TUGAS AKHIR – KI141502

DETEKSI PENYAKIT EPILEPSI BERDASARKAN DATA EEG OTAK MANUSIA DENGAN MENGGUNAKAN *INDEPENDENT COMPONENT ANALYSIS*, *WAVELET TRANSFORM*, DAN *MULTILAYER PERCEPTRON*

ADITYA BAGUSMULYA
NRP 5111100136

Dosen Pembimbing I
Prof. Ir. Handayani Tjandrasa, M.Sc., Ph.D.

Dosen Pembimbing II
Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2015

UNDERGRADUATE THESIS – KI141502

EPILEPSY DISEASE DETECTION BASED ON EEG DATA OF HUMAN'S BRAIN USING INDEPENDENT COMPONENT ANALYSIS, WAVELET TRANSFORM, AND MULTILAYER PERCEPTRON

ADITYA BAGUSMULYA
NRP 5111100136

Supervisor I
Prof. Ir. Handayani Tjandrasa, M.Sc., Ph.D.

Supervisor II
Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2015

LEMBAR PENGESAHAN

DETEKSI PENYAKIT EPILEPSI BERDASARKAN DATA EEG OTAK MANUSIA DENGAN MENGGUNAKAN *INDEPENDENT COMPONENT ANALYSIS, WAVELET TRANSFORM,* DAN *MULTILAYER PERCEPTRON*

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Komputasi Cerdas dan Visi
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh
ADITYA BAGUSMULYA
NRP : 5111 100 136

Disetujui oleh Dosen Pembimbing Tugas Akhir:

1. Prof. Ir. Handayani Tjandrasa, M.Sc., Ph.D. .....
   NIP: 194908231976032001 (Pembimbing 1)

2. Dr. Eng. Chastine Fatichah, S.Kom., M.Kom. .....
   NIP: 197512202001122002 (Pembimbing 2)

SURABAYA
JUNI, 2015

DETEKSI PENYAKIT EPILEPSI BERDASARKAN DATA EEG OTAK MANUSIA DENGAN MENGGUNAKAN *INDEPENDENT COMPONENT ANALYSIS, WAVELET TRANSFORM,* DAN *MULTILAYER PERCEPTRON*

Nama Mahasiswa	**: ADITYA BAGUSMULYA**
NRP	**: 5111100136**
Jurusan	**: Teknik Informatika FTIF-ITS**
Dosen Pembimbing 1	**: Prof. Ir. Handayani Tjandrasa, M.Sc., Ph.D.**
Dosen Pembimbing 2	**: Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.**

Abstrak

Epilepsi merupakan salah satu kelainan pada otak manusia yang tidak dapat disembuhkan. Penyakit ini menimbulkan kejang pada tubuh dan sangat mengganggu aktivitas. Pada tingkat yang parah, epilepsi dapat membahayakan nyawa penderitanya. Oleh sebab itu, epilepsi harus dideteksi secara dini agar penderita segera mendapatkan penanganan yang tepat sehingga keadaannya tidak memburuk..

Pada Tugas Akhir ini, deteksi epilepsi dilakukan dengan menggunakan beberapa metode, yaitu Independent Component Analysis, Wavelet Transform, dan Multilayer Perceptron. Hasil deteksi diklasifikasikan ke dalam tiga kelas, yaitu normal, epilesi tidak kejang, dan epilepsi kejang. Data rekaman electroencephalogram (EEG) *yang digunakan berasal dari "Klinik für Epileptologie, Universität Bonn" yang diperoleh secara online.*

Hasil pendeteksian terbaik dihasilkan dari model yang menggunakan teknik Single Channel Independent Component Analysis pada Independent Component Analysis sebagai

penghilang derau dan ektraksi fitur Discrete Wavelet Transform Daubechies 6 dengan 4 level. Berdasarkan uji coba, metode tersebut menghasilkan akurasi sebesar 92.09%.

Kata kunci : Epilepsi, Independent Component Analysis, Wavelet Transform, Multilayer Perceptron, Klasifikasi

EPILEPSY DISEASE DETECTION BASED ON EEG DATA OF HUMAN'S BRAIN USING INDEPENDENT COMPONENT ANALYSIS, WAVELET TRANSFORM, AND MULTILAYER PERCEPTRON

Student's Name	**: ADITYA BAGUSMULYA**
Student's ID	**: 5111100136**
Department	**: Teknik Informatika FTIF-ITS**
First Advisor	**: Prof. Ir. Handayani Tjandrasa, M.Sc., Ph.D.**
Second Advisor	**: Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.**

Abstract

Epilepsy is one of disorders in human brain that is cannot be healed. This diesease occurs seizuring which bothers patients' activities. In the worst condition, it endangers patients' life. Therefore , the epilepsy must be detected since the early beginning so that patients get a proper treatment immediately for avoiding worse condition.

On this undergraduated thesis, epilepsy detection was build by using three methods; The Independent Component Analysis, Wavelet Transform, and Multilayer Perceptron. The result of detection was classified into 3 classes. They were normal, epilepsy non-seizure, and epilepsy seizure. While the electroencephalogram (EEG) record data used was taken from "Klinik für Epileptologie, Universität Bonn" website.

The best result of classification was achieved by a model that was build by Single Channel Independent Component Analysis technique in Independent Component Analysis, Wavelet Transform, and Multilayer Perceptron as noise removal and Discrete Wavelet Transform using Daubechies 6 with 4 level as feature extraction. Based on test result, the method above obtained an acurracy of 92.09%.

Keywords : Epilepsy, Independent Component Analysis, Wavelet Transform, Multilayer Perceptron, Classification

KATA PENGANTAR

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

Alhamdulillahirabbil'alamin, segala puji bagi Allah SWT, yang telah melimpahkan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul **"DETEKSI PENYAKIT EPILEPSI BERDASARKAN DATA EEG OTAK MANUSIA DENGAN MENGGUNAKAN *INDEPENDENT COMPONENT ANALYSIS, WAVELET TRANSFORM,* DAN *MULTILAYER PERCEPTRON".*** Bagi penulis, pengerjaan Tugas Akhir ini merupakan sebuah pengalaman yang berharga. Selama pengerjaan Tugas Akhir, penulis bisa belajar lebih banyak untuk memperdalam dan meningkatkan apa yang telah didapatkan penulis selama menjalani perkuliahan di Teknik Informatika ITS dan Tugas Akhir ini adalah implementasi dari apa yang telah penulis pelajari.

Selesainya Tugas Akhir ini tidak lepas dari bantuan dan dukungan beberapa pihak. Sehingga pada kesempatan ini penulis mengucapkan syukur dan terima kasih kepada:

1. Allah SWT dan Nabi Muhammad SAW.
2. Keluarga tercinta, Papa, Abuk, Mbak Tia, dan Mbak Ririn yang telah memberikan dukungan, doa, motivasi, dan perhatian yang luar biasa tanpa henti selama penulis mengerjakan Tugas Akhir ini.
3. Ibu Prof. Ir. Handayani Tjandrasa, M.Sc., Ph.D. dan Ibu Dr.Eng. Chastine Fatichah, S.Kom., M.Kom. selaku pembimbing I dan II yang telah membimbing dan membantu penulis serta memberikan motivasi dalam menyelesaikan Tugas Akhir ini.
4. Bapak Dr. Dhany Arifianto ST, M.Eng, dosen Teknik Fisika ITS yang telah memberikan sedikit ilmunya kepada penulis.

5. Ibu Dr. Eng. Nanik Suciati, S.Kom., M.Kom. selaku Kepala Jurusan Teknik Informatika ITS, Bapak Radityo Anggoro, S.Kom.,M.Sc. selaku koordinator TA, Ibu Anny Yuniarti, S.Kom, M.Comp.Sc. selaku dosen wali, dan segenap dosen Teknik Informatika yang telah memberikan ilmunya serta staf karyawan Jurusan Teknik Informatika ITS yang telah memberikan bantuan demi kelancaran admisnistrasi penulis selama kuliah.
6. Petrus, Kinan, Hayam, Rizaldi, Eko Adhi, Dafi, Rizka, Dina, Ruslan, Wilik, dan penghuni Laboraturium IGS dan KCV yang telah menjadi teman seperjuangan dalam menyelesaikan Tugas Akhir dan menjadi tempat bertukar ilmu.
7. Teman-teman Pengurus Harian HMTC 2013-2014 Kabinet Bersahabat dan teman-teman angkatan 2011 yang yang telah berjuang bersama menjalani perkuliahan selama 4 tahun serta menjadi keluarga kedua bagi penulis.
8. Edy, Mbak Merina, dan Mbak Nenny yang telah menemani dan menghibur penulis di saat sedang merasakan kebosanan saat mengerjakan Tugas Akhir ini.
9. Serta semua pihak yang yang telah turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa Tugas Akhir ini masih memiliki banyak kekurangan. Sehingga, penulis mengharapkan kritik dan saran yang membangun dari pembaca untuk perbaikan ke depannya.

Surabaya, Juni 2015

DAFTAR ISI

DAFTAR TABEL

[Halamn ini sengaja dikosongkan]

DAFTAR GAMBAR

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

1.1 Latar Belakang

Epilepsi merupakan salah satu kelainan yang terjadi pada otak manusia. Ciri utama yang tampak adalah penderita mengalami kejang dan kehilangan kesadaran. Kejang ini terjadi minimal dua kali secara tiba-tiba tanpa sebab baik ringan maupun berat [1]. Penyebab secara pasti dari kelainan ini belum diketahui, namun sebagian besar dari para penderita memiliki riwayat epilepsi dari keluarga mereka sehingga bisa disebut dengan penyakit akibat genetis. Selain itu epilepsi juga bisa disebabkan oleh penyakit lain yang menyerang otak. Epilepsi bisa terjadi pada manusia di segala umur. Namun, gejalanya sangat terlihat pada anak-anak dan lansia di atas 65 tahun.

Epilepsi secara medis tidak dapat disembuhkan. Obat hanya mampu mengontrol frekuensi terjadinya kejang dan juga tingkat keparahan dari kejang tersebut. Namun perlu diwaspadai, pada tingkat yang parah epilepsi bisa mengakibatkan kematian bagi penderitanya. Selain dari sisi fisik, dampak negatif yang ditimbulkan dari epilepsi bagi penderitanya yaitu psikis dan kehidupan sosial. Secara psikis, penderita epilepsi akan merasa tidak nyaman, gelisah bahkan depresi karena kejang yang terus-menerus dialaminya. Sementara itu, kerugian yang diterima penderita dalam kehidupan sosialnya yaitu adanya diskriminasi. Penderita tidak akan diizinkan mengendarai kendaraan karena dapat membahayakan diri sendiri maupun orang lain apabila secara tiba-tiba terjadi kejang saat berkendara. Lebih jauh lagi, di beberapa negara, penderita epilepsi dilarang menikah [2]. Untuk meminimalisir terjadinya hal-hal yang merugikan penderita maka epilepsi perlu dideteksi sejak dini dan segera diberikan perawatan

yang tepat sehingga penderita bisa hidup layaknya manusia normal.

Teknologi selalu berkembang setiap saat. Tidak dapat dipungkiri, saat ini kehidupan manusia serba dimudahkan dengan adanya teknologi begitu juga dengan dunia kedokteran. Pengaplikasian yang sangat berguna yaitu adanya tekologi untuk mendeteksi penyakit atau kelainan pada tubuh manusia. Salah satu kelainan yang dapat dideteksi dengan teknologi yaitu epilepsi. Sebelumnya, diagnosis dilakukan secara manual yaitu dengan mengamati rekaman sinyal *electroencephalogram* (EEG) dari otak. Cara konvensional seperti ini membutuhkan waktu yang cukup lama dan kurang efisien. Sehingga dibuatlah pendeteksi otomatis data EEG yang diolah sedemikian rupa. Beberapa penelitian sudah pernah dilakukan dengan menggunakan metode-metode yang berbeda, baik *preprocessing* maupun klasifikasi, diantaranya menggunakan algoritma *Fuzzy* dan *Support Vector Machine* [3] dan *Permutation Entropy* dan *Support Vector Machine* [4]. Pada penggunaan *Permutation Entropy* dan *Support Vector Machine* menghasilkan rata-rata keakuratan sebesar 84.18% dengan hanya menggunakan 2 kelas saja.

Berdasarkan hasil akurasi di atas, pada tugas akhir ini akan dibangun sebuah perangkat lunak yang sama namun menggunakan metode yanng berbeda. Metode yang akan diimplementasikan yaitu *Independent Component Analysis* dan *Wavelet Transform* sebagai preprocessing dan *Multilayer Preceptron* sebagai klasifikasi. Data yang digunakan sebagai masukan yaitu data rekam EEG otak manusia. Diharapkan nilai akurasi pendeteksian bisa lebih baik dari penelitian yang sudah pernah dilakukan. Hal ini tentu akan membuat deteksi menjadi lebih tepat sehingga penderita epilepsi bisa mendapatkan perawatan yang sesuai.

1.2 Rumusan Masalah

Berikut ini adalah beberapa rumusan masalah yang diangkat dalam pembuatan tugas akhir ini :

1. Bagaimana membangun perangkat lunak untuk mendeteksi epilepsi dengan mengimplementasikan *Independent Component Analysis* (ICA) dan *Wavelet Transform* (WT) dan *Multilayer Perceptron* (MLP)?
2. Bagaimana menghilangkan derau pada data sinyal EEG dengan menggunakan ICA?
3. Bagaimana melakukan ekstraksi fitur data sinyal EEG dengan menggunakan WT?
4. Bagaimana melakukan klasifikasi fitur-fitur data sinyal EEG dengan menggunakan MLP untuk mendeteksi epilepsi?
5. Bagaimana melakukan uji performa perangkat lunak dengan menggunakan Confusion Matrix?

1.3 Batasan Masalah

Batasan masalah yang ditetapkan untuk pembuatan tugas akhir ini adalah sebagai berikut :

1. Dataset yang digunakan adalah data rekam sinyal EEG manusia normal dan epilesi yang diambil dari *Klinik fur Epileptologie Universitat Bonn*.
2. Pembangunan perangkat lunak menggunakan Matlab R2013a.
3. Ada tiga hasil klasifikasi, yaitu normal, epilepsi tidak kejang, dan epilepsi kejang.

1.4 Tujuan

Tujuan pembuatan dari tugas akhir ini yaitu :

1. Membangun sebuah perangkat lunak yang mampu melakukan deteksi epilepsi pada manusia melalui data rekam sinyal EEG otak.
2. Mengimplementasikan ICA untuk menghilangkan derau pada data rekam sinyal EEG.
3. Mengimplementasikan WT untuk melakukan ekstraksi fitur dari data rekam sinyal EEG.
4. Mengimplementasikan MLP untuk melakukan klasifikasi hasil ekstraksi fitur.

5. Melakukan uji perfoma terhadap perangkat lunak pendeteksi epilesi yang telah dibangun.

1.5 Manfaat

Manfaat yang diperoleh dari pembuatan tugas akhir ini adalah adanya kemudahan dalam melakukan diagnosis bagi dunia kedokteran apakah seseorang menderita epilepsi atau tidak. Semakin cepat diketahui maka semakin cepat pula penanganan dan pengobatan yang diberikan sehingga penderita bisa hidup selayaknya manusia normal.

1.6 Metodologi

Tahapan-tahapan yang dilakukan dalam pengerjaan Tugas Akhir ini adalah sebagai berikut:

1. Penyusunan proposal Tugas Akhir.
   Proposal tugas akhir ini berisi rencana tugas akhir yang akan dikerjakan sebagai syarat untuk menyelesaikan studi dan meraih gelar Strata-1 Teknik Informatika. Terdapat penjelasan mengenai latar belakang pengambilan tema, rumusan masalah, batasan masalah, tujuan, dan manfaat dari rencana tugas akhir ini. Selain itu juga dijelaskan mengenai metode apa saja yang digunakan serta penjelasannya. Agar lebih mudah dipahami, penjelasan disertai dengan diagram alir.

2. Studi literatur
   Beberapa literatur yang perlu dipelajari lebih dalam lagi untuk membangun perangkat lunak pendeteksi epilepsi ini yaitu *Independent Component Analysis*, *Wavelet Transform*, *Multilayer Perceptron*, *K-Fold Cross Validation*, *Confusion Matrix*, dan Matlab.

3. Analisis dan desain perangkat lunak

   Untuk membangun perangkat lunak pendeteksi epilepsi ini harus melalui beberapa tahap yaitu menghilangkan derau dari data rekam sinyal EEG sehingga data bersih dari sinyal lain. Lalu melakukan ekstraksi fitur dari sinyal yang telah bersih agar data yang berupa sinyal dapat diklasifikasi. Terakhir adalah melakukan klasifikasi untuk mendapatkan model yang tepat. Selanjutnya perangkat lunak bisa digunakan untuk melakukan pendeteksian epilepsi.

4. Implementasi perangkat lunak

   Perangkat lunak ini akan dibangun dengan menggunakan bahasa pemrograman dan kakas bantu Matlab R2013a dengan fungsi yang sudah tersedia di dalamnya.

5. Pengujian dan evaluasi

   Proses pengujian perangkat lunak ini nantinya menggunakan teknik *K-Fold Cross Validation* dengan K bernilai 10. Artinya, semua data baik normal maupun epilepsi akan disatukan dan diacak. Selanjutnya dari keseluruhan data dibagi menjadi 10. Keseluruhan data akan menjadi data latih dan data uji secara bergantian. Jika data 1 menjadi data pengujian maka data 2 hingga 10 menjadi data pembelajaran. Jika data 2 menjadi data pengujian maka data 1 dan 3 hingga 10 menjadi data pembelajaran dan seterusnya. Evaluasi perangkat lunak menggunakan teknik *Confusion Matrix* dengan mengukur *accuracy*, *sensitivity*, dan *specificity*.

6. Penyusunan buku Tugas Akhir.

   Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat.

1.7 Sistematika Penulisan Laporan Tugas Akhir

Buku Tugas Akhir ini merupakan laporan secara lengkap mengenai Tugas Akhir yang telah dikerjakan baik dari sisi teori, rancangan, maupun implementasi sehingga memudahkan bagi pembaca dan juga pihak yang ingin mengembangkan lebih lanjut. Sistematika penulisan buku tugas akhir secara garis besar antara lain:

Bab I Pendahuluan

Bab ini berisi latar belakang, rumusan masalah, batasan masalah dan tujuan pembuatan Tugas Akhir. Selain itu, metodologi pengerjaan dan sistematika penulisan laporan Tugas Akhir juga terdapat di dalamnya.

Bab II Dasar Teori

Bab ini berisi penjelasan mengenai teori-teori yang digunakan dalam pengerjaan Tugas Akhir secara rinci.

Bab III Perancangan Perangkat Lunak

Bab ini berisi penjelasan tentang rancangan dari sistem yang akan dibangun. Rancangan ini dituliskan dalam bentuk *pseudocode*.

Bab IV Implementasi

Bab ini berisi implementasi dari rancangan yang telah dibuat pada bab sebelumnya. Implementasi disajikan dalam bentuk *code* secara keseluruhan disertai dengan penjelasannya.

Bab V Uji Coba Dan Evaluasi

Bab ini berisi pengujian sistem dari segi *accuracy*, *sensitivity*, dan *specificity* dengan berdasarkan pada skenario yang telah ditentukan.

Bab VI Kesimpulan Dan Saran

Bab ini berisi kesimpulan dari hasil pengujian sistem pada bab sebelumnya. Selain itu juga terdapat saran yang ditujukan untuk pengembangan sistem.

[Halaman ini sengaja dikosongkan]

BAB II
TINJAUAN PUSTAKA

Dalam bab ini akan dijelaskan mengenai teori-teori yang merupakan dasar dari pembangunan sistem. Selain itu terdapat penjelasan yang menunjang pengerjaan Tugas Akhir ini sehingga dapat memberikan gambaran secara umum sistem yang akan dibangun.

2.1 *Independent Component Analysis*

Independent Component Analysis (ICA) adalah sebuah metode statistik yang dapat menemukan faktor-faktor tersembunyi yang ada dalam sekumpulan variabel acak atau sinyal [5] . ICA mampu memisahkan sinyal-sinyal yang saling bertumpangan menjadi beberapa sinyal yang mempunyai karakteristik statistik yang berbeda [6]. Metode ini termasuk dalam *Blind Source Separation* yaitu menemukan sumber-sumber sinyal yang tidak teramati karena saling bertumpangan. Sinyal yang dapat dipisahkan dengan ICA antara lain suara dan sinyal kedipan mata pada EEG. Contohnya yaitu pada kasus "cocktail party" di mana dua orang berhitung secara bersamaan dengan menggunakan *microphone* yang berbeda sehingga suara keduanya menjadi satu sinyal. Model dari ICA dapat dilihat pada Persamaan 2.1.

$$x = A.s \tag{2.1}$$

Dimana $x = [x_1, x_2, \ldots x_n]^T$ adalah sinyal campuran yang akan dipisahkan komponen-komponennya. Vektor x ini adalah hasil perkalian dari matriks penyampur A dan $s = [s_1, s_2, \ldots s_m]^T$ yang merupakan vektor independen. Namun, kedua komponen tersebut tidak diketahui. Sehingga perlu dibentuk sebuah persamaan untuk memprediksi nilai s. Persamaan tersebut seperti pada Persamaan 2.2.

$$u = W.x \tag{2.2}$$

Dimana $u = [u_1, u_2, \ldots u_m]^T$ adalah vektor prediksi dari *s* dan *W* adalah matriks pemisah yang merupakan invers dari matriks *A*. Pada dasarnya ICA adalah metode yang digunakan untuk sinyal *multi channel*, namun dapat dimodifikasi untuk sinyal *single channel* [7].

2.1.1 *Single Channel Independent Component Analysis* (SCICA)

SCICA adalah salah satu metode hasil pengembangan dari ICA yang ditujukan bagi sinyal yang hanya memiliki *channel* tunggal [8]. Metode ini diawali dengan membagi sinyal menjadi beberapa bagian dan mengubahnya menjadi sebuah matriks.

$$x(k) = [x(k\tau), \ldots, x(k\tau + N - 1)]^T \tag{2.3}$$

$$X = [x(1), \ldots, x(K)]^T \tag{2.4}$$

Simbol $x(t)$ adalah sinyal asli. *x(k)* sinyal yang telah dibagi. N adalah panjang sinyal masing-masing bagian. Matriks *X* adalah gabungan semua bagian sinyal *x(k)* untuk *k* =1, 2, ..., *K*. Simbol τ merupakan panjang waktu tiap bagian dan $(K\tau + N - 1)$ panjang sinyal asli.

Pada matriks *X* dilakukan ICA kemudian dari proses tersebut dihasilkan matriks pencampuran dan pemisahan *A* dan *W*. Selanjutnya, untuk mendapatkan sinyal yang bersih derau dengan cara mengalikan *X* dengan *W*. Terakhir kembalikan ke bentuk sinyal asli dengan menggabungkan kembali menjadi satu sinyal tunggal.

2.1.2 *Wavelet Independent Component Analysis* (WICA)

WICA mengubah *single channel* menjadi *multi channel* dengan cara mendekomposisi sinyal menjadi beberapa level

dengan *Stationary Discrete Wavelet Transformation* [7]. Pemilihan tipe *mother wavelet* yang digunakan tergantung dengan bentuk sinyal. Setelah dilakukan dekomposisi sebanyak level yang diinginkan, selanjutnya dilakukan ICA pada sinyal tersebut. Hasilnya adalah berupa matriks pencampuran dan pemisahan. Matriks ini kemudian dikalikan dengan sinyal dekomposisi sehingga menghasilkan sinyal yang bersih derau. Sinyal tersebut dikembalikan ke bentuk asal dengan cara invers wavelet.

2.2 *Wavelet Transform*

Wavelet Transform adalah metode yang digunakan untuk melokalisasi suatu fungsi dalam ruang dan skala. Wavelet dikembangkan berdasarkan metode yang sudah ada sebelumnya yaitu *Fourier Transform*. Ide utamanya yaitu mengekspresikan sinyal sebagai kombinasi linear dari sebuah fungsi tertentu yang diperoleh dengan menggeser sebuah induk wavelet dan mengonvolusikannya dengan sinyal tersebut [9]. Hasil dari konvolusi ini akan menghasilkan kumpulan koefisien yang disebut dengan koefisien wavelet. Terdapat berbagai jenis wavelet diantaranya yaitu *discrete* dan *stationary discrete wavelet*.

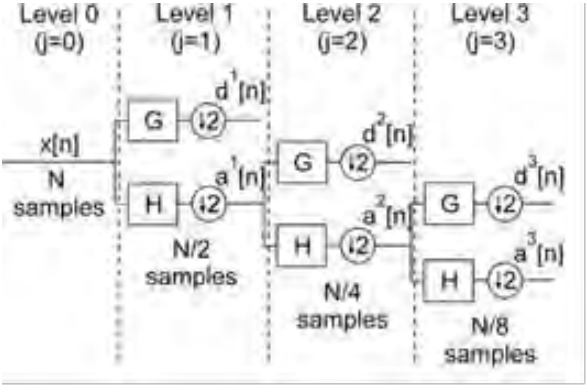
2.2.1 *Discrete Wavelet Transform* (DWT)

DWT digunakan untuk mendekomposisi melalui teknik filterisasi [10]. Sinyal dilewatkan pada filter yang memiliki frekuensi dan skala yang berbeda. Terdapat dua jenis filter yaitu *highpass filter* dan *lowpass filter*. Persamaan dari kedua filter tersebut terdapat pada Persamaan 2.5 dan 2.6 berikut ini.

$$Y_{high}[k] = \sum_n X[n]h[2k - n] \tag{2.5}$$

$$Y_{low}[k] = \sum_n X[n]g[2k - n] \tag{2.6}$$

Dimana Y_{high} adalah hasil dari *highpass filter* atau disebut dengan *Detail* (D) dan Y_{low} adalah hasil dari *lowpass filter* atau *Approximation* (A). Proses dekomposisi ini diulang berkali-kali pada *approximation* sesuai dengan jumlah level yang dikendaki dan menghasilkan D dan A baru. Setiap dekomposisi berarti mengurangi lebar pita frekuensi menjadi setengah sinyal sebelumnya (*downsampling*). Contoh hasil dekomposisi terdapat pada Gambar 2.1.



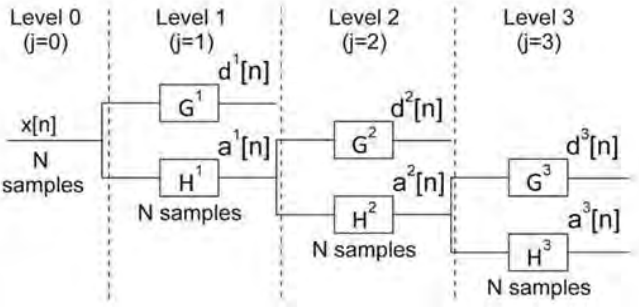
Gambar 2. 1 Dekomposisi dengan DWT **[11]**

Pada gambar di atas tampak sinyal *x[n]* didekomposisi menjadi $d^1[n]$ atau D1 dan $a^1[n]$ atau A1. Artinya, hasil dekomposisi masih pada level 1. Selanjutnya pada A1 dilakukan dekomposisi lagi sehingga menjadi $d^2[n]$ atau D2 dan $a^2[n]$ atau A2. Jika dilakukan wavelet diskrit dengan 4 level maka hasil akhir koefisien waveletnya yaitu D1, D2, D3, D4, dan A4.

2.2.2 *Stationary Discrete Wavelet Transformation* (SWT)

Stationary Discrete Wavelet Transformation (SWT) atau biasa disebut dengan *Undecimate Wavelet Transform* adalah modifikasi dari DWT. Hal yang menjadi pembeda adalah pada SWT panjang sinyal transformasi yang dihasilkan setiap levelnya

tetap atau tidak adanya *downsampling*. Seperti yang telah dijelaskan pada sub bab sebelumnya, pemilihan *mother wavelet* yang digunakan disesuaikan secara manual dengan bentuk sinyal masukan. Gambar 2.2 menunjukkan proses dekomposisi satu dimensi dengan SWT.



Gambar 2. 2 Proses dekomposisi SWT [11]

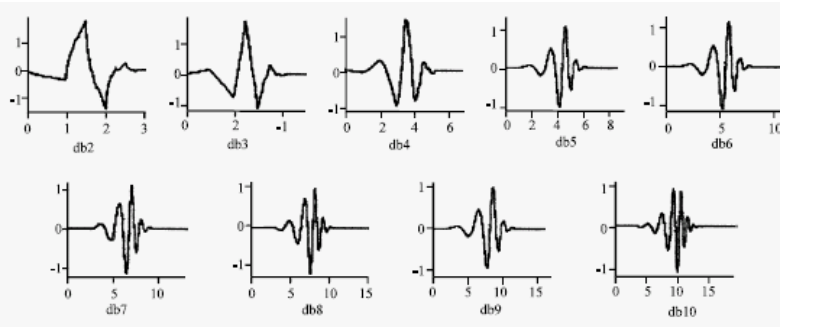
Tipe *mother wavelet* terdiri dari berbagai macam *family,* diantaranya Haar, Daubechies, Coiflet, dan Symlet. Setiap proses Wavelet Transform memerlukan satu *mother wavelet dan* pemilihannya dilakukan secara manual dengan mengamati bentuk sinyal dari sinyal. Bentuk yang paling mendekati adalah filter yang dianggap terbaik untuk digunakan. *Mother wavelet* yang digunakan pada Tugas Akhir ini, baik SWT maupun DWT, yaitu Daubechies *family*. Daubechies memiliki sepuluh jenis filter seperti yang ditunjukkan pada Gambar 2.3.

2.3 *Multilayer Perceptron Neural Network*

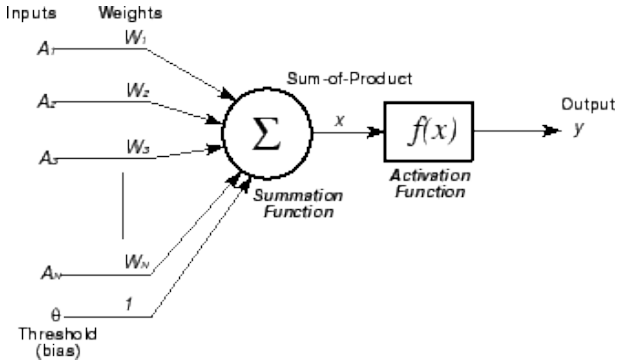
2.3.1 *Artificial Neural Network*

Artificial Neural Network (ANN) atau Jaringan Syaraf Tiruan (JST) merupakan suatu algoritma klasifikasi yang diadopsi dari sistem kerja syaraf makhuk hidup. ANN terdiri dari komputasi dari banyak *neuron* yang saling terhubung. Setiap penghubung dari

neuron memiliki bobot tertentu selain itu setiap satu *neuron* dipengaruhi oleh bias. Gambar 2.4 menunjukkan contoh dari struktur ANN.



Gambar 2. 3 Macam-macam filter pada Daubechies *family* **[12]**



Gambar 2. 4 Struktur ANN **[13]**

Struktur di atas terdiri dari dua lapisan. Lapisan pertaman yaitu masukan yang disimbolkan dengan *A*. Jumlah masukan dimungkinkan lebih dari satu. Bobot disimbolkan dengan *W*. Masing-masing masukan memiliki nilai bobot. Selain bobot, terdapat nilai bias. Bias merepresentasikan nilai kesalahan. Lapisan

kedua yaitu keluaran. Keluaran yang dihasilkan ANN hanya terdiri dari dua kelas saja.

2.3.2 *Binary Sigmoid*

Untuk menentukan hasil klasifikasi, ANN membutuhkan fungsi aktivasi. Fungsi aktivasi memiliki jenis yang beragam. Salah satunya yang sering digunakan adalah *binary sigmoid*. Persamaan dari fungsi aktivasi *binary sigmoid* sesuai dengan Persamaan 2.7.

$$f(x) = \frac{1}{1 + e^{-1}} \tag{2.7}$$

Sesuai namanya, hasil dari fungsi ini berkisar antara 0 hingga 1.

2.3.3 *Multilayer Perceptron Neural Network*

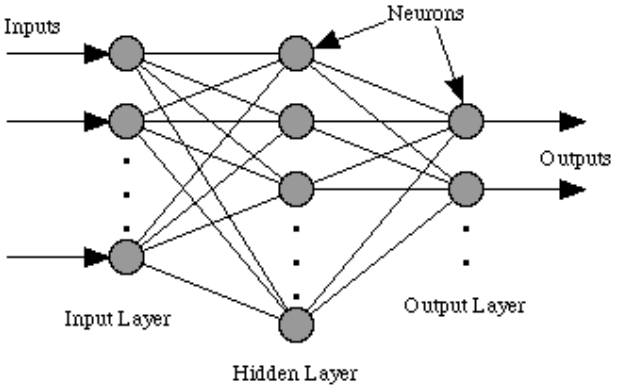
Kasus klasifikasi yang memiliki lebih dari dua kelas dapat diselesaikan dengan menggunakan *Multilayer Perceptron Neural Network* (MLP). Lapisan yang terdapat dalam struktur MLP lebih dari tiga atau lebih. Strukturnya tampak seperti Gambar 2.5.

MLP memiliki bobot dan bias seperti ANN. Namun yang membedakan dengan ANN yaitu adanya lapisan tersembunyi atau *hidden layer* di antara masukan dan keluaran. Jumlah lapisan tersembunyi bisa saja lebih dari satu.

2.3.4 Algoritma *Backpropagation*

Algoritma *Backpropagation* merupakan algoritma yang bisa diterapkan pada MLP. Algoritma ini terdiri dari dua tahap, yaitu *feedforward* dan *backpropagation.* Pada saat *feedforward,* sinyal masukan dihitung oleh semua neuron di setiap lapisan sehingga menghasilkan keluaran tanpa melakukan perubahan bobot.

Sedangkan pada *backpropagation,* semua bobot diperbarui berdasarkan kesalahan jaringan. Kedua tahap ini diulang hingga bobot membuat keluaran yang sama atau mendekati hasil yang sebenarnya.



Gambar 2. 5 Struktur MLP **[14]**

Pseudocode dari algoritma ini adalah sebagai berikut [15] :

1. Buat sebuah *network* dengan n_{in} unit *input*, n_{hidden} unit tersembunyi, dan n_{out} unit *output.*
2. Inisialisasi semua bobot dan bias ke angka random yang kecil
3. Sampai kondisi terminasi tercapai, kemudian untuk tiap data *training*, lakukan:
   - Masukkan data *training* sebagai *input* dan hitung *output* jaringan O_u
   - Untuk tiap unit *output* k, hitung term *error* δ_k
     $\delta_k \leftarrow O_k (1 - O_k) (t_k - O_k)$
   - Untuk tiap unit tersembunyi hitung term *error* δ_h
     $\delta_h \leftarrow O_h (1- O_h) \sum_{k \in output\ s} w_{h,k}\ \delta k$
   - Ubah tiap bobot dan bias w_{ij}
     $W_{ij} \leftarrow Wij + \Delta Wij$

Di mana
$\Delta Wij \leftarrow \eta\ \delta_j x_{ij}$

Keterangan:
- δ_k adalah *error output*
- O_k adalah keluaran dari lapisan *output*
- t_k adalah *output* yang diharapkan
- δ_h adalah *error* pada lapisan tersembunyi
- O_h adalah keluaran dari lapisan tersembunyi
- $W_{h,k}$ adalah bobot antara lapisan *output* dan lapisan tersembunyi
- w_{ij} adalah semua bobot dan bias pada jaringan saraf
- η adalah *learning rate*
- x_{ij} adalah keluaran dari lapisan
- ΔWij adalah selisih bobot saat ini dengan bobot sebelumnya

2.4 *K-Fold Cross Validation*

K-Fold Cross Validation adalah sebuah metode untuk menguji suatu perangkat lunak dengan sejumlah dataset. Pada *K-Fold Cross Validation* membagi data menjadi *K* data set yang bebas [16]. Sejumlah *K*-1 data digunakan sebagai data latih sedangkan sisanya menjadi data uji. Pengujian ini diulangi sebanyak *K* kali hingga semua set pernah menjadi data uji dan data latih. Nilai akurasi yang sudah didapat kemudian dihitung rata-ratanya sebagai nilai akurasi akhir.

2.5 *Confusion Matrix*

Confusion Matrix (CM) adalah matriks yang mengandung informasi tentang kelas sebenarnya dan prediksi yang dihasilkan oleh sistem klasifikasi [17]. CM tidak hanya untuk klasifikasi 2 kelas, namun juga untuk 3 kelas atau lebih. Pada Tugas Akhir ini

ada 3 kelas pada klasifikasi. Tabel 2.1 adalah tabel yang merepresentasikan matriks dari CM 3 kelas [18].

Tabel 2. 1 Tabel *Confusion Matrix*

		Predicted class		
		A	B	C
Actual class	A	tp_A	e_{AB}	e_{AC}
	B	e_{BA}	tp_B	e_{BC}
	C	e_{CA}	e_{CB}	tp_C

Nilai *tp* adalah jumlah data uji yang kelas prediksinya sama dengan kelas yang sebenarnya. Sedangkan *e* adalah jumlah kelas prediksi yang tidak sesuai dengan kelas sebenarnya.

Beberapa nilai evaluasi yang bisa dihitung berdasarkan matriks tersebut yaitu *accuracy, sensitivity*, dan *specificity*. A*ccuracy* adalah proporsi jumlah prediksi yang terklasifikasi secara tepat. *Sensitivity* adalah perbandingan dari jumlah data *tp* terhadap kelas yang terprediksi berlabel kelas negatif. *Specificity* adalah perbandingan dari jumlah data *tp* terhadap kelas yang terprediksi positif. Karena terdapat 3 kelas sehingga masing-masing kelas memiliki *sensitivity* dan *specificity*. Persamaannya dapat dilihat pada Persamaan 2.8,2.9,dan 2.10.

$$Accuracy = \frac{tp_A + tp_B + tp_C}{n} \tag{2.8}$$

$$Sensitivity_{class} = \frac{tp_{class}}{tp_{class} + fn_{class}} \tag{2.9}$$

$$Specificity_{class} = \frac{tn_{class}}{tn_{class} + fp_{class}} \tag{2.10}$$

Pada perhitungan *accuracy*, *n* adalah jumlah seluruh data pengujian. Pada *sensitivity* dan *specificity*, *fn* adalah jumlah false negatif dimana data yang sebenarnya negatif namun terprediksi positif. Misal pada kelas A, nilai fn_A adalah jumlah e_{AB} dan e_{AC}. Nilai fp_A adalah jumlah e_{BA} dan e_{CA}. Sedangkan nilai tn_A adalah jumlah tp_B, e_{BC}, e_{CB}, dan tp_C.

2.6 Normalisasi

Normalisasi adalah sebuah proses untuk mengubah suatu data ke dalam rentang nilai tertentu. Tujuannya adalah untuk menghindari pesebaran data yang terlalu jauh sehingga sebuah variabel tidak mendominasi terhadap variabel lain. Salah satu jenis normalisasi adalah normalisasi skala. Pada normalisasi, skala rentang yang umum digunakan yaitu 0 hingga 1. Rumus umum normalisasi skala adalah sebagai berikut [19] :

$$X_{new} = \frac{(X - X_{min})(X_{newmax} - X_{newmin})}{X_{max} - X_{min}} + X_{newmin} \quad (2.11)$$

Variabel X adalah nilai yang akan dinormalisasi. Variabel X_{max} dan X_{min} adalah nilai tertinggi dan terendah pada nilai-nilai atribut di mana X berada. Variabel X_{newmax} dan X_{newmin} adalah nilai tertinggi dan terendah yang diinginkan, misalkan 1 dan 0. Apabila rentang yang digunakan adalah -1 hingga 1 maka rumus di atas menjadi seperti berikut :

$$X_{new} = \frac{(X - X_{min}) + 2}{X_{max} - X_{min}} - 1 \quad (2.12)$$

[Halaman ini sengaja dikosongkan]

BAB III
DESAIN PERANGKAT LUNAK

Pada bab ini akan dijelaskan mengenai rancangan sistem perangkat lunak yang akan dibuat. Perancangan yang dijelaskan meliputi data dan proses. Data yang dimaksud adalah data yang akan diolah dalam perangkat lunak baik digunakan sebagai pembelajaran maupun pengujian sehingga tujuan Tugas Akhir ini bisa tercapai. Proses yaitu tahap-tahap yang ada dalam sistem sebagai pengolah data meliputi *Independent Component Analysis*, *Wavelet Transform*, dan *Multilayer Perceptron*.

3.1 Data

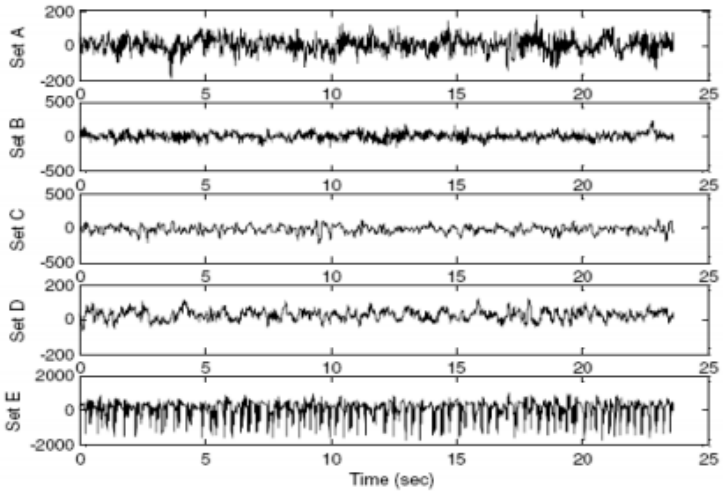
Pada sub bab ini akan dijelaskan mengenai data yang digunakan sebagai masukan perangkat lunak untuk selanjutnya diolah dan dilakukan pengujian sehingga menghasilkan data keluaran yang diharapkan

3.1.1 Data Masukan

Data masukan adalah data yang digunakan sebagai masukan awal dari sistem. Data yang digunakan dalam perangkat lunak deteksi penyakit epilepsi dengan Menggunakan *Independent Component Analysis, Wavelet Transform,* dan *Multilayer Perceptron* adalah data sinyal otak manusia (EEG) yang diunduh dari website milik "Klinik für Epileptologie, Universität Bonn". Data ini terdiri dari 5 set rekaman EEG (A-E) yang mana setiap set berisi 100 data sinyal otak *single-channel* masing-masing data berdurasi 23,6 detik dan dengan frekuensi 173.61 Hz serta 4097 nilai. Kelima data set tersebut diambil dari orang-orang yang berbeda dengan kondisi yang berbeda pula. Keseluruhan data telah diperiksa ada tidaknya derau secara kasat mata.

Perekaman data A dan B diambil dari sukarelawan yang sehat namun dengan kondisi yang berbeda. A direkam dengan mata

terbuka sedangkan B dengan mata tertutup. Data C, D, dan E merupakan data sinyal otak dari pasien penderita epilepsi, namun sama halnya dengan A dan B, perekaman dilakukan dalam kondisi yang berbeda. Data D diambil dari bagian kepala yang berada dalam zona kejang sedangkan C pada formasi hippocampal belahan kepala yang lainnya. Keduanya direkam dalam keadaan tidak kejang. Data E merupakan sinyal otak pasien yang sedang dalam keadaan kejang. Berdasarkan penjelasan di atas dapat disimpulkan bahwa ada 200 data sinyal otak manusia sehat, 200 data epilepsi tidak kejang, dan 100 data epilepsi kejang. Contoh sinyal dari masing-masing data set mulai A hingga E secara berurutan ditunjukkan pada Gambar 3.1.



Gambar 3. 1 Contoh data masukan masing-masing data set. [9]

3.1.2 Data Keluaran

Data masukan akan diproses dengan menggunakan metode *Independent Component Analysis, Wavelet Transform*, dan *Multilayer Perceptron*. Pada metode klasifikasi, *Multilayer Perceptron*, data akan dibagi menjadi dua yaitu data pembelajaran

dan pengujian. Hasil dari proses klasifikasi tersebut adalah nama kelas dan nilai-nilai performa yang mencakup *accuracy, specificity,* dan *sensitivity*.

3.2 Desain Umum Sistem

Rancangan perangkat lunak deteksi penyakit epilepsi dengan menggunakan *Independent Component Analysis*, *Wavelet Transform*, dan *Multilayer Perceptron* dimulai dengan melakukan *preprocessing* yaitu menghilangkan derau dari kelima data sinyal otak manusia dengan menggunakan *Independent Component Analysis*. Ada dua jenis *Independent Component Analysis* yang digunakan yaitu *Single Channel* dan *Wavelet*. Hasil dari preprocessing tersebut adalah sinyal otak yang sudah bersih dari derau. Sinyal-sinyal tersebut digunakan sebagai masukan untuk tahap selanjutnya.

Tahap selanjutnya adalah ekstraksi fitur dari masing-masing sinyal dengan menggunakan metode *Discrete Wavelet Transform*. Satu rekaman sinyal otak manusia yang telah bersih dari derau dibagi menjadi beberapa bagian kemudian setiap bagian akan didekomposisi menggunakan *Discrete Wavelet Transform* sebanyak jumlah level. Hasil dekomposisi akan menghasilkan fitur-fitur yang diperoleh dari perhitungan nilai rata-rata, minimum, maksimum, dan standar deviasi.

Keluaran dari tahap di atas adalah data set baru yang siap untuk masukan tahap selanjutnya yaitu *Multilayer Perceptron*. Data dibagi menjadi data pembelajaran dan pengujian sehingga dapat diperoleh nilai performa dari hasil klasifikasi. Dari data pembelajaran akan dihasilkan sebuah model yang digunakan untuk pengujian performa. Pengujian performa dari model menggunakan metode *K-Fold Cross Validation* seperti yang telah dijelaskan pada bab sebelumnya dengan parameter yang berbeda-beda sesuai dengan skenario. Diagram alir desain umum perangkat lunak ditunjukkan pada Gambar 3.2 berikut ini.

Gambar 3. 2 Diagram alir rancangan perangkat lunak secara umum

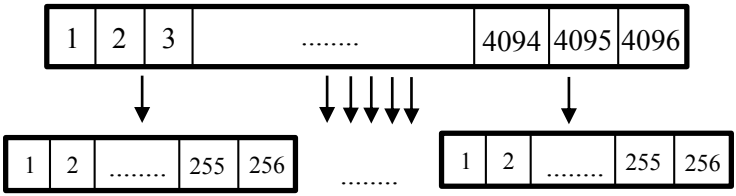
3.3 Preprocessing

Data yang telah diambil tidak bisa langsung digunakan dalam perangkat lunak. Oleh karena itu, perlu dilakukan *preprocessing* untuk sedikit mengubah data sehingga data yang

akan digunakan memiliki kualitas yang lebih baik. Dalam pengerjaan Tugas Akhir ini *preprocessing* yang dilakukan adalah menghilangkan derau dari sinyal otak EEG. Secara garis besar, penghilangan derau menggunakan *Independent Component Analysis* (ICA). Pada Tugas Akhir ini metode ICA yang dilakukan akan menggunakan sebuah program *open source* berbasis Matlab yaitu EEGLAB. Ada dua macam ICA yang akan digunakan sebagai perbandingan yaitu *Single Channel* dan *Wavelet*.

3.3.1 Penghilangan Derau dengan *Single Channel* ICA

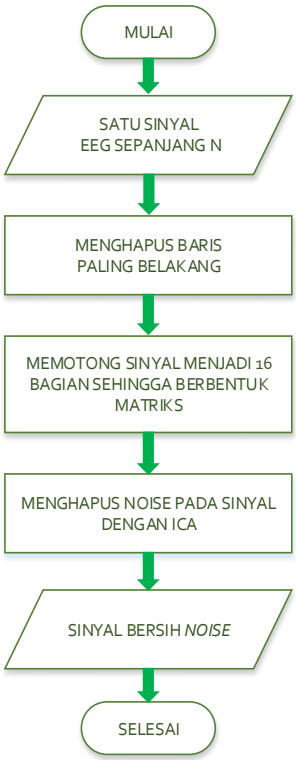
Prinsip penghilangan derau dengan *Single Channel* ICA (SCICA) adalah dengan mengubah *single channel* menjadi *multi channel*. Satu sinyal yang memiliki 4097 baris nilai dibagi menjadi 16 bagian sama panjang. Sehingga masing-masing bagian memiliki 256 baris nilai. Oleh karena itu pembagian sinyal menyisakan satu nilai. Satu nilai ini bisa diabaikan. Lalu hasil pembagian disusun menjadi sebuah matriks sehingga seolah-olah satu sinyal tadi menjadi sebuah sinyal EEG *multi channel* yang memiliki 16 *channel* dan masaing-masing 256 baris nilai. Ilustrasi pembagian ditunjukkan pada Gambar 3.3 di bawah ini.



Gambar 3. 3 Pembagian sinyal EEG menjadi beberapa bagian

Setelah dilakukan proses pengubahan bentuk data seperti penjelasan di atas, data baru tersebut digunakan sebagai input dari proses ICA untuk menghilangkan derau dengan menggunakan EEGLAB. Hasil dari ICA adalah sinyal EEG yang berukuran sama

dengan input namun dalam keadaan bersih dari derau. Gambar 3.4 adalah diagram alir dari proses SCICA.



Gambar 3. 4 Diagram alir penghilangan derau dengan SCICA

3.3.2 Penghilangan Derau dengan *Wavelet* ICA

Sama seperti metode *Single Channel* ICA, metode *Wavelet* ICA (WICA) juga mengubah sinyal *single channel* menjadi *multi channel* sehingga metode ICA bisa diterapkan untuk menghilangkan derau. Mula-mula sebuah sinyal dihapus nilai paling belakangnya sehingga memiliki 4096 baris nilai. Kemudian dilakukan dekomposisi sinyal dengan menggunakan *Stationary*

Wavelet Transform (SWT) dengan tipe wavelet Daubechies 2 sebanyak 4 level. SWT digunakan sebagai pengubah sinyal dari *single channel* ke *multi channel,* karena hasil dekomposisi yang dihasilkan ukurannya tetap atau tidak terjadi *downsampling*. Hal ini sesuai dengan masukkan yang dibutuhkan untuk proses penghilangan noise dengan ICA.

Transformasi dengan SWT menghasilkan 5 buah sinyal dekomposisi sepanjang 4096 baris nilai yang terdiri dari 4 detail dan 1 approksimasi. Kelima sinyal ini dianggap sebagai sebuah sinyal EEG yang *multi channel*. Selanjutnya, dilakukan penghilangan derau dengan metode ICA pada EEGLAB dan tetap menghasilkan 5 buah sinyal. Untuk mengembalikan sinyal ke bentuk aslinya, kelima sinyal tersebut direkonstruksi menjadi satu sinyal bersih derau dengan menggunakan invers dari *Stationary Wavelet Transform*. Gambar 3.5 merupakan diagram alir dari WICA.

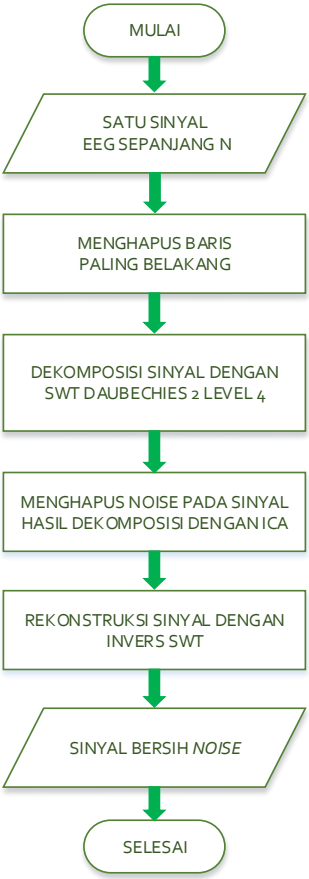
3.4 *Processing*

Pada sub bab ini akan dijelaskan mengenai proses pengolahan data sehingga menghasilkan keluaran yang diharapkan seperti yang telah dijelaskan sebelumnya. Ada dua metode yang digunakan yaitu *Discrete Wavelet Transform* dan *Multilayer Perceptron*. *Discrete Wavelet Transform* digunakan untuk mengekstraksi fitur-fitur dari setiap sinyal. Fitur-fitur tersebut digunakan sebagai masukan dari proses klasifikasi *Multilayer Perceptron* baik proses pembelajaran maupun pengujian sehingga sinyal tersebut bisa dikelompokkan sebagai sinyal otak manusia normal, epilepsi tidak kejang, atau epilepsi kejang.

3.4.1 Ekstraksi Fitur dengan *Discrete Wavelet Transform*

Setelah menghasilkan sinyal yang bersih dari derau, tahap selanjutnya adalah pengekstraksian fitur. Sebuah sinyal keluaran dari *preprocessing* mula-mula dipotong-potong sepanjang 256

baris nilai sehingga menghasilkan 16 bagian sama seperti saat preprocessing dengan SCICA.



Gambar 3. 5 Diagram alir penghilangan derau dengan WICA

Selanjutnya, pada setiap bagian akan dilakukan dekomposisi dengan menggunakan *Discrete Wavelet Transform*.

Tipe *mother wavelet* yang digunakan sama seperti pada *preprocessing* di atas yaitu Daubechies, namun tipe dan level sesuai dengan skenario uji coba. Jumlah level yang digunakan ada 4 macam, yaitu level 4, 5, 6, dan 7. Pada setiap levelnya, jumlah fitur yang dihasilkan berbeda-beda seperti yang ditunjukkan pada Tabel 3.1.

Tabel 3. 1 Jumlah fitur yang dihasilkan pada setiap level dekomposisi

Jumlah Level Dekomposisi	Jumlah Fitur
4	20
5	24
6	28
7	32

Sebagai gambaran, digunakan Daubechies 2 dengan 4 level. Hasil dekomposisi adalah 5 buah sinyal *subband* yang terdiri dari 4 detail dan 1 approksimasi. Pada setiap level, jumlah baris nilai akan berkurang setengah dari level sebelumnya ini dikarenakan pada DWT terjadi pengurangan lebar pita frekuensi sehingga panjang tiap hasil dekomposisi akan terus berkurang. Hal inilah yang membedakan SWT dan DWT. DWT tidak mungkin digunakan sebagai pengubah bentuk sinyal seperti SWT karena input dari ICA adalah sebuah matriks. Tabel 3.2 menampilkan panjang baris nilai hasil dari dekomposisi. Kelima *subband* hasil dekomposisi disebut koefisien wavelet di mana koefisien-koefisien tersebut mengandung informasi penting pada sinyal.

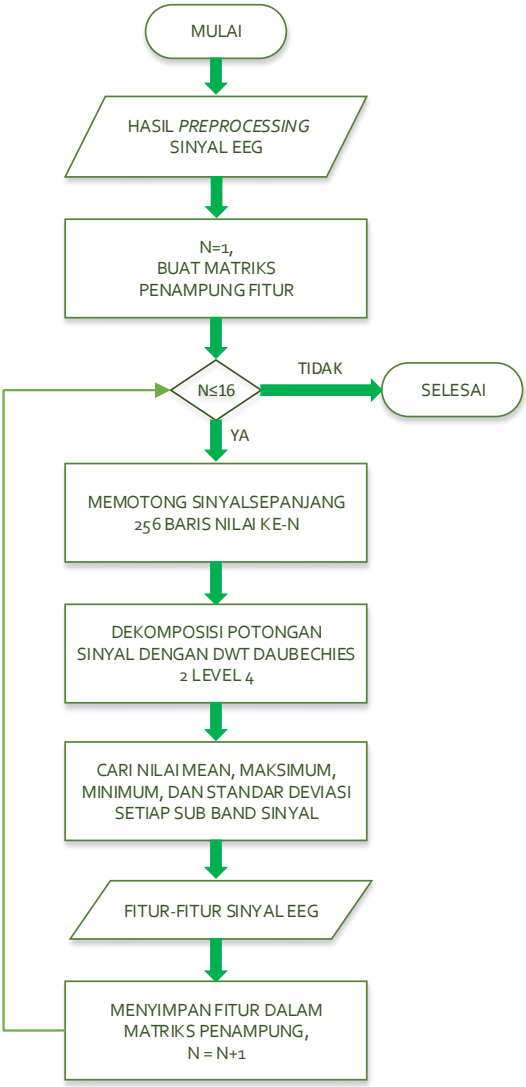
Tabel 3. 2 Hasil dekomposisi dengan DWT

Level Dekomposisi	Panjang Sinyal
Detail Level 1	2048
Detail Level 2	1024
Detail Level 3	512
Detail Level 4	256
Aproksimasi Level 4	256

Setiap *subband* hasil dekomposisi akan dihitung empat nilai statistik sebagai repersentasi dari setiap frekuensi. Nilai statistik yang dihitung antara lain nilai mean, maksimum, minimum, dan standar deviasi. Sehingga pada setiap satu bagian menghasilkan 20 nilai statistik yang dijadikan fitur. Seluruh fitur dalam sebuah data set dijadikan satu ke dalam matriks penampung. Setiap satu bagian disimpan dalam matriks penampung sebagai satu baris baru. Sehingga, jika satu sinyal awal dibagi menjadi 16 dan jumlah sinyal ada 20 maka total baris data yang ada dalam matriks penampung adalah 320 baris. Jumlah keseluruhan data dari 5 data set yaitu 1600 data. Proses ekstraksi fitur ini ditunjukkan pada diagram alir Gambar 3.6.

3.4.2 Klasifikasi dengan *Multilayer Perceptron*

Proses sebelumnya menghasilkan data set baru yaitu ekstraksi fitur-fitur dari sinyal. Data set tersebut selanjutnya masuk ke proses klasifikasi dengan menggunakan *Multilayer Perceptron* algoritma *Backpropagation*. Ada dua tahap dalam klasifikasi ini, yaitu tahap pembelajaran dan tahap pengujian.

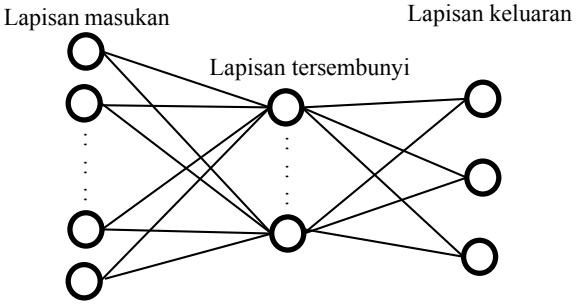


Gambar 3. 6 Diagram alir ekstraksi fitur dengan DWT

3.4.2.1 Tahap Pembelajaran

Pada sub bab sebelumnya telah dijelaskan tentang data masukan yaitu 5 data set (A-E) sukarelawan baik normal maupun pasien epilepsi. Sebelum memulai proses pembelajaran, isi data set dijadikan satu file kemudian diacak secara random agar semua kelas yang ada tercampur rata sehingga model yang dihasilkan lebih bagus.

Dalam Tugas Akhir ini *Multilayer Perceptron* yang digunakan hanya terdiri dari satu lapisan tersembunyi, jadi terdapat tiga bagian yaitu lapisan masukan, lapisan tersembunyi, dan lapisan keluaran. Gambar 3.7 menunjukkan rancangan jaringan syaraf yang digunakan.



Gambar 3. 7 Rancangan jaringan syaraf dalam proses klasifikasi

Mula-mula inisialisasi bobot dan bias secara random serta *learning rate* dan jumlah *epoch*. Jumlah node lapisan tersembunyi yaitu 15. Jumlah tersebut mengambil salah satu dari *Rules of Thumb* yaitu nilai di antara dari jumlah node masukan dan keluaran **[20]**. *Learning rate* yang digunakan sebesar 0.1 dan *epoch* 200. Sebelum proses klasifikasi, data harus dinormalisasi terlebih dahulu agar sebaran nilai data menjadi seragam dan tidak ada fitur yang mendominasi selain itu juga menyederhanakan kalkulasi. Selanjutnya, data pembelajaran satu per-satu dimasukkan ke proses *feedfoward* dan dihasilkan nilai pada lapisan keluaran. Keluaran yang dihasilkan, baik dari lapisan masukan ke lapisan tersembunyi

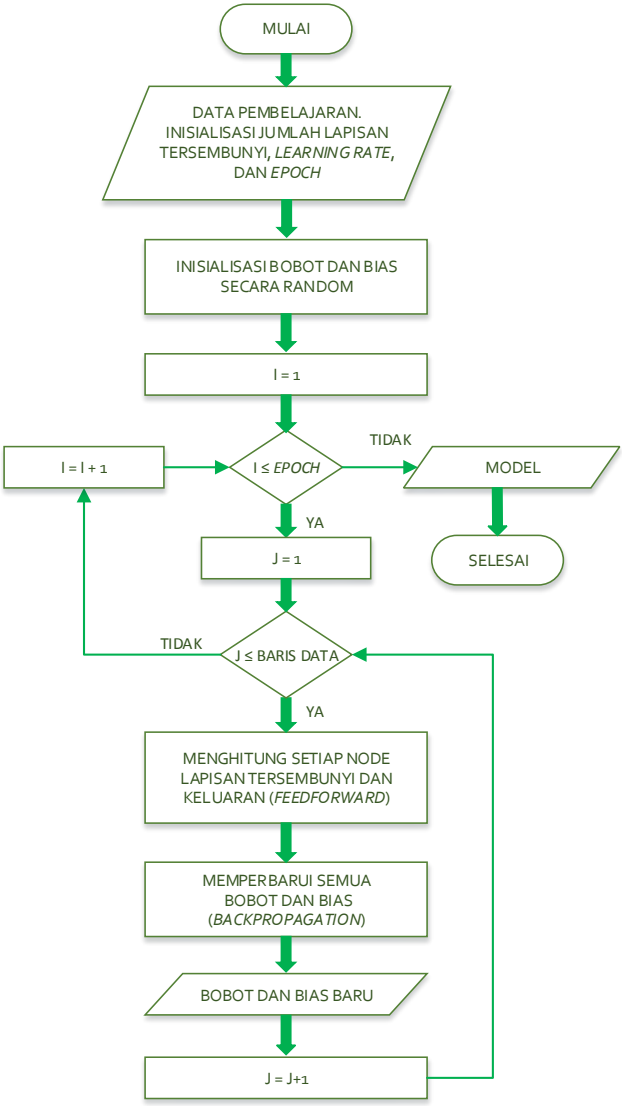
ataupun lapisan tersembunyi ke lapisan keluaran, selalu dihitung nilai aktivasi sigmoidnya. Setelah itu dilakukan proses *backpropagation* untuk memperbarui nilai bobot dan bias yang digunakan saat proses *feedforward.* Nilai bobot dan bias yang baru digunakan dalam proses *feedforward* dan *backpropagation* data selanjutnya. Seterusnya langkah ini diulang hingga data pembelajaran habis dan jumlah *epoch* terpenuhi. Diagram alir proses pembelajaran ditunjukkan pada Gambar 3.8.

Hasil akhir dari proses pembelajaran ini adalah nilai bobot dan bias yang terbaik atau disebut model sehingga saat proses pengujian model ini mampu menglasifikasikan data pengujian dengan tepat.

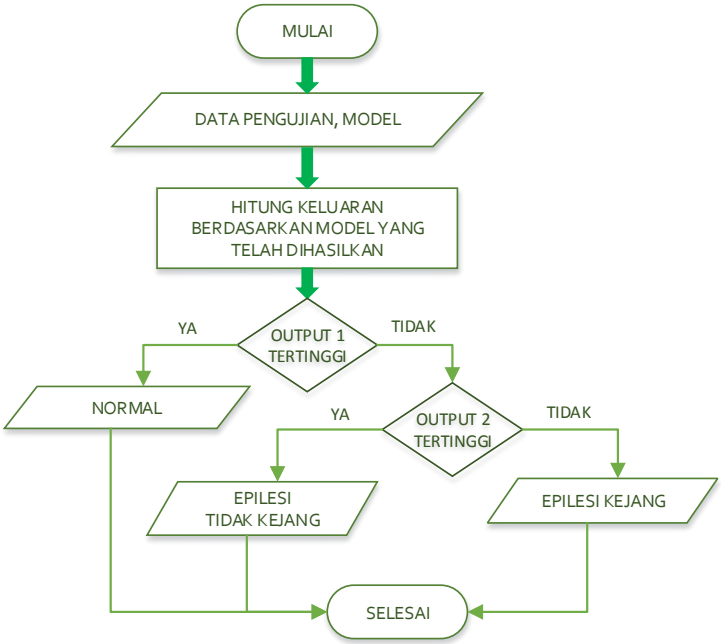
3.4.2.2 Tahap Pengujian

Untuk tahap pengujian diperlukan data pengujian sesuai dengan data set yang digunakan dalam tahap pembelajaran apakah menggunakan SCICA atau WICA. Sama seperti tahap pembelajaran, data perlu dinormalisasi terlebih dahulu dengan menggunakan Persamaan 2.12. Tahap ini hanya melakukan sekali perhitungan yaitu bagian *feedforward* tentu saja dengan fungsi aktivasi sigmoid juga.

Hasil keluaran akan menentukan apakah data tersebut tergolong normal, epilepsi tidak kejang, atau epilepsi kejang. Penentuan kelas didasarkan pada nilai tertinggi dari ketiga keluaran. Jika keluaran 1 yang tertinggi maka data uji tersebut termasuk dalam kelas 1 atau normal begitu pula untuk kelas 2 dan 3. Gambar 3.9 menunjukkan diagram alir tahap pengujian.



Gambar 3. 8 Diagram alir tahap pembelajaran *Multilayer Perceptron* algoritma *Backpropagation*

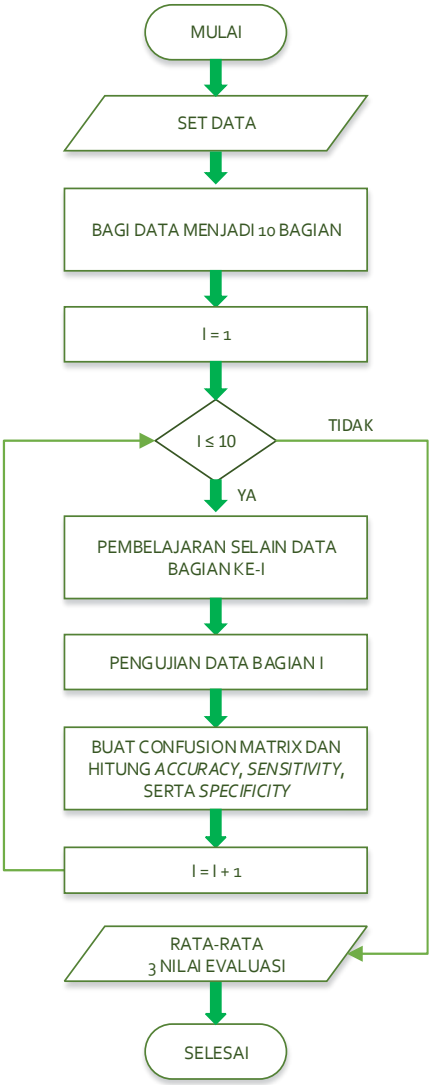


Gambar 3. 9 Diagram alir tahap pengujian model *Multilayer Perceptron*

3.5 Uji Performa

Tahap ini bertujuan untuk mengukur seberapa baik metode-metode yang telah digunakan pada *preprocessing* dan *processing*. Uji performa pada Tugas Akhir ini menggunakan metode *K-Fold Cross Validation* dengan *K* adalah 10. Data set dibagi menjadi 10 bagian data sama banyak. Jika bagian 1 menjadi data pengujian maka bagian 2 hingga 9 menjadi data pembelajaran. Sedangkan jika bagian 2 menjadi data pengujian maka bagian 1 dan 3 hingga 9 menjadi data pembelajaran. Begitu seterusnya hingga bagian 10 menjadi data pengujian.

Setiap pengujian dihitung nilai *Confusion Matrix*-nya sehingga bisa diperoleh *accuracy*, *sensitivity*, dan *specificity*. Nilai Jika *K* adalah 10 ini maka ada 10 nilai evaluasi di atas yang kemudian dihitung rata-ratanya. Nilai evaluasi tersebut merepresentasikan performa dari metode yang telah dilakukan. Diagram alir uji performa ditunjukkan pada Gambar 3.10.



Gambar 3. 10 Diagram alir uji performa *dengan K-Fold Cross Validation* untuk $K = 10$

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI

Pada bab ini akan dijelaskan mengenai implementasi dari rancangan yang telah dibahas pada Bab III, baik fungsi utama maupun fungsi lainnya.

4.1 Lingkungan Implementasi

Pada tabel 4.1. ini adalah spesifikasi dari lingkungan implementasi yang digunakan untuk membangun perangkat lunak.

Tabel 4. 1 Spesifikasi lingkungan implementasi

Perangkat	Spesifikasi
Perangkat keras	Prosesor: Intel® Pentium® 2020M 2.40GHz Memori: 6.00 GB
Perangkat lunak	Sistem Operasi: Microsoft Windows 8.1 64-bit Perangkat Pengembang: Matlab R2013a Perangkat Pembantu: Microsoft Excel 2013

4.2 Implementasi

Sub bab implementasi ini menjelaskan tentang pembangunan perangkat lunak secara detail termasuk menampilkan kode program bila diperlukan mulai *preprocessing* hingga uji performa. Sebelumnya perlu diketahui bahwa dalam Tugas Akhir ini, setiap data set A hingga E diambil 20 file acak rekaman sinyal otak agar pemrosesan lebih sederhana.

4.2.1 Implementasi *Preprocessing*

4.2.1.1 Implementasi Pembagian Sinyal

Seperti yang telah dijelaskan pada sub bab sebelumnya bahwa pada penghapusan derau dengan SCICA diperlukan adanya pengubahan bentuk pada setiap rekaman sinyal dari bentuk vektor menjadi sebuah matriks. Berikut ini adalah kode program cutSignal.m yang digunakan untuk membagi sinyal sehingga menjadi matriks yang siap dijadikan sebagai masukkan EEGLAB.

```
1   s1 = [];
2   awal = 1;
3   fileName = 'S099.txt';
4   fileID = fopen(fileName,'r');
5   val = fscanf(fileID, '%f');
6   val = val';
7   for j = 1:16
8        temp = val(1,awal:awal+255);
9        s1 = [s1; temp];
10       awal = awal+256;
11  end
```

Kode Sumber 4. 1 Kode program pemotongan sinyal pada sebuah file rekaman sinyal

Kode Sumber 4.1 di atas membaca folder dan file .txt yang berisi rekaman sinyal otak sesuai dengan data set. Setiap satu sinyal dibagi menjadi 16 bagian sehingga menghasilkan matriks berukuran 16x256 yang disimpan dalam sebuah array sehingga terdapat 20 array untuk tiap data set. Untuk mempermudah proses ekstraksi fitur, 20 array ini disimpan dalam sebuah array besar berukuran 320x256.

4.2.1.2 Implementasi Dekomposisi Sinyal dengan SWT

Pada WICA sebuah sinyal harus diubah menjadi matriks sehingga perlu adanya dekomposisi sebanyak sejumlah level. Pada Tugas Akhir ini sinyal akan didekomposisi sebanyak 4 level

dengan Daubechies 2 sehingga tiap sinyal menjadi matriks berukuran 5x4096. Kode Sumber 4.2 di bawah ini adalah kode program swtSignal.m untuk mendekomposisi sinyal.

```
%membaca sinyal raw WICA
list = dir('S');

signal = [];

for i = 3:length(list)
    fileName = fullfile('S',list(i).name);
    fileID = fopen(fileName,'r');
    val = fscanf(fileID, '%f');
    val = val';
    signal = [signal; val];
end

signal(:,size(signal,2)) = [];

a1 = swt(signal(1,:),4,'db2');
a2 = swt(signal(2,:),4,'db2');
a3 = swt(signal(3,:),4,'db2');
a4 = swt(signal(4,:),4,'db2');
a5 = swt(signal(5,:),4,'db2');
a6 = swt(signal(6,:),4,'db2');
a7 = swt(signal(7,:),4,'db2');
a8 = swt(signal(8,:),4,'db2');
a9 = swt(signal(9,:),4,'db2');
a10 = swt(signal(10,:),4,'db2');
a11 = swt(signal(11,:),4,'db2');
a12 = swt(signal(12,:),4,'db2');
a13 = swt(signal(13,:),4,'db2');
a14 = swt(signal(14,:),4,'db2');
a15 = swt(signal(15,:),4,'db2');
a16 = swt(signal(16,:),4,'db2');
a17 = swt(signal(17,:),4,'db2');
a18 = swt(signal(18,:),4,'db2');
a19 = swt(signal(19,:),4,'db2');
a20 = swt(signal(20,:),4,'db2');
```

Kode Sumber 4. 2 Kode program membaca dan mendekomposisi sinyal dengan SWT

Program akan membaca folder di mana data set berada kemudian satu per satu file rekaman dibaca dan dimasukkan ke dalam sebuah array. Agar hasilnya genap 4096 baris nilai, baris

paling terakhir dihapus. Baris 16 hingga akhir adalah proses dekomposisi setiap sinyal 4 level dengan menggunakan fungsi *swt* yang telah disediakan oleh Matlab. *db2* adalah *mother wavelet* Daubechies 2. Setiap hasil dekomposisi disimpan dalam array a1 hingga a20 sesuai jumlah file rekaman yang digunakan. Penyimpanan secara terpisah ini bertujuan untuk memudahkan untuk proses selanjutnya, yaitu ICA dengan EEGLAB.

4.2.1.3 Implementasi ICA dengan EEGLAB

Setiap teknik pengubahan bentuk sinyal, baik SCICA maupun WICA, pada akhirnya akan menggunakan ICA sebagai penghilang derau. Oleh karena itu, tiap array hasil kedua proses diatas dimsukkan ke dalam EEGLAB sebagai data masukkan kemudian dilakukan ICA. Hasil dari proses tersebut adalah matriks pemisah yang nantinya digunakan untuk memperoleh sinyal yang bersih derau. Untuk WICA, setelah sinyal bersih didapatkan perlu adanya proses rekonstruksi agar sinyal kembali menjadi *single channel* dengan menggunakan fungsi *iswt* pada Matlab. Kode Sumber 4.3 berikut ini adalah kode program proses rekonstruksi yang terdapat di dalam pascaICA.m.

```
1  weight = EEG.icasphere*EEG.icaweights;
2  s1 = inv(weight)*s1;
3  s1 = iswt(s1,'db2');
```

Kode Sumber 4. 3 Kode program proses rekonstruksi

Hasil rekonstruksi tiap data pada masing-masing data set disimpan dalam file *.csv*.

4.2.2 Implementasi *Processing*

4.2.2.1 Ekstraksi Fitur dengan DWT

Setelah ICA selesai dijalankan dan menghasilkan sinyal yang bersih dari derau, tahap selanjutnya yaitu mengekstraksi fitur dari sinyal sebagai masukan dari tahap klasifikasi. Ekstraksi fitur hasil dari *preprocessing* dengan SCICA maupun WICA dilakukan dengan menggunakan fungsi *dwt* Daubechies.

Implementasi ekstraksi fitur preprocessing dengan SCICA dalam waveleting2.m ditunjukkan pada Kode Sumber 4.4.

```
1   %ekstraksi fitur scica
2   signal = csvread('ICAO1.csv');
3   [row, col] = size(signal);
4
5
6   fileN = 'WAVO1';
7   fileCatName = strcat(fileN,'.csv');
8   fileID1 = fopen(fileCatName,'w');
9
10  for i=1:row
11          temp = signal(i,:);
12          [App, Det1] = dwt(temp, 'db2');
13          [App, Det2] = dwt(App, 'db2');
14          [App, Det3] = dwt(App, 'db2');
15          [App, Det4] = dwt(App, 'db2');
16          %level 5
17          %[App, Det5] = dwt(App, 'db2');
18          %level 6
19          %[App, Det6] = dwt(App, 'db2');
20          %level 7
21          %[App, Det7] = dwt(App, 'db2');
22
23          [meann, maximum, minimum, stddeviation] =
    features(Det1);
24          fprintf(fileID1, '%f, %f, %f, %f, ', meann,
    maximum, minimum, stddeviation);
25          [meann, maximum, minimum, stddeviation] =
    features(Det2);
26          fprintf(fileID1, '%f, %f, %f, %f, ', meann,
    maximum, minimum, stddeviation);
27          [meann, maximum, minimum, stddeviation] =
    features(Det3);
28
```

```
        fprintf(fileID1, '%f, %f, %f, %f, ', meann,
29  maximum, minimum, stddeviation);
        [meann, maximum, minimum, stddeviation] =
30  features(Det4);
        fprintf(fileID1, '%f, %f, %f, %f, ', meann,
31  maximum, minimum, stddeviation);
        [meann, maximum, minimum, stddeviation] =
    features(App);
33      fprintf(fileID1, '%f, %f, %f, %f, ', meann,
    maximum, minimum, stddeviation);
34      %level 5
35      %[meann, maximum, minimum, stddeviation] =
    features(Det6);
36      %fprintf(fileID1, '%f, %f, %f, %f, ', meann,
    maximum, minimum, stddeviation);
37      %level 6
38      %[meann, maximum, minimum, stddeviation] =
    features(App);
39      %fprintf(fileID1, '%f, %f, %f, %f, ', meann,
    maximum, minimum, stddeviation);
40      %level 7
41      %[meann, maximum, minimum, stddeviation] =
    features(App);
42      %fprintf(fileID1, '%f, %f, %f, %f, ', meann,
    maximum, minimum, stddeviation);
43      fprintf(fileID1, '0,0,1');
44      fprintf(fileID1, '\n');
45  end
46  fclose(fileID1);
```

Kode Sumber 4. 4 Kode program ekstraksi fitur sinyal hasil SCICA

Pada kode program di atas, hasil ICA dari data set O atau B dibaca kemudian dilakukan ekstraksi fitur dan hasilnya disimpan dalam file WAVO1.csv. Penghilangan derau dengan SCICA menghasilkan sinyal berupa matriks berukuran 16x256 sebanyak 20 array. Seperti yang telah dijelaskan pada sub bab sebelumnya, 20 array ini disatukan menjadi sebuah array besar agar memudahkan proses ekstraksi. Sinyal ini setiap barisnya, 1x265, didekomposisi 4 level. Baris 10 hingga 45 adalah proses ekstraksi fitur. Baris 12 hingga 21 adalah proses dekomposisi dengan DWT. Untuk 4 level hanya dibutuhkan kode baris ke 12 hingga 15. Sehingga menghasilkan 4 detail (Det1, Det2, Det3, Det4) dan 1

aproksimasi (App). Dekomposisi dengan DWT menghasilkan setengah jumlah baris nilai dari level sebelumnya. Berikut ini Tabel 4.1 adalah hasil dekomposisi untuk tiap baris sinyal dengan 4 level.

Tabel 4. 2 Hasil dekomposisi DWT 4 level

Level Dekomposisi	Panjang Sinyal
Detail Level 1	128
Detail Level 2	64
Detail Level 3	32
Detail Level 4	16
Aproksimasi Level 4	16

Baris 17, 19, dan 21 dijalankan ketika akan membuat data set dengan DWT 5 level, 6 level, dan level 7. Kedua data set ini dibutuhkan untuk kebutuhan uji coba sebagai pembanding.

Setiap *subband* dihitung 4 nilai statistik yang digunakan sebagai fitur yaitu mean, maksimum, minimum, dan standar deviasi. Penghitungan nilai-nilai ini menggunakan fungsi features() DWT 4 level seperti pada baris 23, 25, 27, 29, dan 31 . Setiap satu bagian sinyal menghasilkan 20 fitur dan disimpan dalam sebuah file *.csv*. Baris 43 menandakan kelas dari data. Karena terdapat 3 kelas, maka tertulis 3 digit nilai. (1,0,0) untuk kelas 1, (0,1,0) untuk kelas 2, dan (0,0,1) untuk kelas 3. Apabila semua baris sudah tereksekusi maka menghasilkan matriks berukuran 320x23 untuk setiap data set.

Sama seperti baris 17, 19, dan 21, baris 35 hingga 42 dijalankan ketika membuat data set dengan DWT 5 level, 6 level, dan level 7.

Untuk *preprocessing* dengan WICA, secara umum sama dengan proses di atas, hanya saja perlu dilakukan pembagian sinyal. Setiap satu sinyal sepanjang 4096 baris nilai dibagi menjadi 16. Setelah itu setiap bagian didekomposisi seperti proses di atas. Berikut ini Kode Sumber 4.5 adalah kode program waveleting.m

untuk ekstraksi fitur sinyal hasil penghilangan derau dengan WICA.

```
%ekstraksi fitur wica
signal = csvread('ICAN.csv');
[row, col] = size(signal);

fileN = 'WAVN';
fileCatName = strcat(fileN,'.csv');
fileID1 = fopen(fileCatName,'w');

for i=1:row
    awal = 1;
    for j = 1:16
        temp = signal(i,awal:awal+255);
        [App, Det1] = dwt(temp, 'db2');
        [App, Det2] = dwt(App, 'db2');
        [App, Det3] = dwt(App, 'db2');
        [App, Det4] = dwt(App, 'db2');
        %level 5
        %[App, Det5] = dwt(App, 'db2');
        %level 6
        %[App, Det6] = dwt(App, 'db2');
        %level 7
        %[App, Det7] = dwt(App, 'db2');

        [meann, maximum, minimum, stddeviation] = features(Det1);
        fprintf(fileID1, '%f, %f, %f, %f, ', meann, maximum, minimum, stddeviation);
        [meann, maximum, minimum, stddeviation] = features(Det2);
        fprintf(fileID1, '%f, %f, %f, %f, ', meann, maximum, minimum, stddeviation);
        [meann, maximum, minimum, stddeviation] = features(Det3);
        fprintf(fileID1, '%f, %f, %f, %f, ', meann, maximum, minimum, stddeviation);
        [meann, maximum, minimum, stddeviation] = features(Det4);
        fprintf(fileID1, '%f, %f, %f, %f, ', meann, maximum, minimum, stddeviation);
        [meann, maximum, minimum, stddeviation] = features(App);
        fprintf(fileID1, '%f, %f, %f, %f, ', meann, maximum, minimum, stddeviation);
        %level 5

```

```
        %[meann, maximum, minimum, stddeviation] =
36  features(Det6);
        %fprintf(fileID1, '%f, %f, %f, %f, ', meann,
37  maximum, minimum, stddeviation);
38      %level 6
        %[meann, maximum, minimum, stddeviation] =
39  features(Det7);
        %fprintf(fileID1, '%f, %f, %f, %f, ', meann,
40  maximum, minimum, stddeviation);
41      %level 7
        %[meann, maximum, minimum, stddeviation] =
42  features(App);
        %fprintf(fileID1, '%f, %f, %f, %f, ', meann,
43  maximum, minimum, stddeviation);
44      fprintf(fileID1, '0,1,0');
45      fprintf(fileID1, '\n');
46      awal = awal+256;
47    end
48  end
    fclose(fileID1);
```

Kode Sumber 4. 5 Kode program ekstraksi fitur sinyal hasil WICA

```
1   %menghitung fitur-fitur
2   function [meann, maximum, minimum, stddeviation] =
3   features(inputVec);
4
5       meann = mean(inputVec);
6       maximum = max(inputVec);
7       minimum = min(inputVec);
8       stddeviation = std(inputVec);
```

Kode Sumber 4. 6 Kode program fungsi features()

Sinyal disimpan dalam sebuah array berukuran 20x4096. Setiap baris diambil kemudian dibagi menjadi 16 bagian seperti pada baris 11. Kemudian proses selanjutnya dekomposisi dan menghitung nilai statistik sebagai fitur seperti yang ditunjukkan pada Kode Sumber 4.6. Pada akhir proses dihasilkan 320 baris data untuk setiap data set masing-masing data sepanjang 23 nilai.

4.2.2.2 Klasifikasi dengan MLP

Tahap terakhir setelah ekstraksi fitur yaitu klasifikasi. Pada tahap ekstraksi fitur telah dihasilkan 10 data set baru yaitu masing-masing 5 data set A, B, C, D, dan E untuk SCICA dan selebihnya WICA. Pada tahap ini dibangun sebuah program yang akan menghasilkan sebuah model terbaik untuk proses pengujian.

4.2.2.2.1 Fungsi mainMlp()

Fungsi ini adalah fungsi utama dari proses pembelajaran di mana semua fungsi-fungsi lainnya seperti *feedforward* dan *backpropagation* dipanggil. Berikut ini adalah kode program dari mainMlp().

```
inputData = csvread('ABCDE_WICA.csv');
[r1, c1] = size(inputData);
x = 160;
awal = 1;
akurasi = [];
sensitivity = [];
specificity = [];
hiddenLayer = 15;
learningRate = 0.1;
epoch = 200;
tic;
for i=1:10
    inputDataTemp = inputData;
    inputDataTemp2 = inputDataTemp(:,1:c1-3);
    inputDataTemp2 = normalz(inputDataTemp2);
    inputTestN = inputDataTemp2(awal:awal+159,1:c1-3);
    outputTest = inputDataTemp(awal:awal+159,c1-2:c1);
    inputDataTemp(awal:awal+159,:) = [];
    inputDataTemp2(awal:awal+159,:) = [];
    inputMlpN = inputDataTemp2(:,1:c1-3);
    outputMlp = inputDataTemp(:,c1-2:c1);

    inputWeight = rand(hiddenLayer, c1-3);
    hiddenWeight = rand(3,hiddenLayer);
    bias1 = rand(1, hiddenLayer);
    bias2 = rand(1, 3);

   for ep=1:epoch
       for j=1:r1-x
```

```
30              inputRow = inputMlpN(j,:);
31              outputRow = outputMlp(j,:);
32              [outputInputZj, outputHiddenYk] =
    feedForward(hiddenLayer,inputRow,inputWeight,hiddenWeig
    ht, bias1, bias2);
33              [deltaInput, deltaHiddenWjk, deltabias1,
    deltabias2] =
    backPropagation(learningRate,inputRow,outputInputZj,out
    putHiddenYk,outputRow,hiddenWeight,bias1,bias2);
34              [inputWeightNew, hiddenWeightNew, newBias1,
    newBias2] = updateWeightBias(inputWeight,hiddenWeight,
    bias1, bias2, deltaInput, deltaHiddenWjk, deltabias1,
    deltabias2);
35              inputWeight = inputWeightNew;
36              hiddenWeight = hiddenWeightNew;
37              bias1 = newBias1;
38              bias2 = newBias2;
39          end
40      end
41
42          [resultMlp] = mainTestMlp(hiddenLayer, inputTestN,
    inputWeight, hiddenWeight, bias1, bias2);
43          [acc, spec, sens, confusionM] =
    validate(resultMlp, outputTest);
44          akurasi = [akurasi; acc*100];
45          sensitivity = [sensitivity; sens*100];
46          specificity = [specificity; spec*100];
47          awal = awal+160;
48  end
49
50  accmean = mean(akurasi)
51  specymeanA = mean(specificity(:,1))
52  specymeanB = mean(specificity(:,2))
53  specymeanC = mean(specificity(:,3))
54  sensymeanA = mean(sensitivity(:,1))
55  sensymeanB = mean(sensitivity(:,2))
56  sensymeanC = mean(sensitivity(:,3))
57
58  tm = toc;
```

Kode Sumber 4. 7 Kode program fungsi mainMlp()

Pada Tugas Akhir ini, data pembelajaran dan pengujian disimpan dalam file berformat *.csv* sehingga perangkat lunak harus bisa membaca file tersebut dan Matlab sudah menyediakan fungsi *csvread* seperti pada baris 1. Selanjutnya data dipisahkan antara data fitur dan target. Sebelum dilakukan proses pembelajaran,

nilai-nilai fitur ini perlu dinormalisasi dengan fungsi normalz() seperti pada baris 15. Fungsi rand() berguna untuk menentukan nilai random bobot dan bias. Baris 28 hingga 40 adalah proses pembelajaran yang diulang sebanyak *epoch*. Di dalamnya terdapat pemanggilan fungsi feedForward(), backPropagation(), dan updateWeightBias().

4.2.2.2.2 Fungsi normalz()

Fungsi ini digunakan untuk menormalisasi data dalam rentang tertentu. Pada Tugas Akhir ini data dinormalisasi dalam rentang -1 hingga 1. Rumus yang digunakan seperti pada Persamaan 2.12 di Bab II. Berikut ini adalah kode program dari fungsi normalz().

```
1  %normalisasi data range -1 - 1
2  function inputMlpN = normalz(inputMlp)
3      inputMlpN = [];
4      cols = size(inputMlp,2);
5      for i=1:cols
6          mi = min(inputMlp(:,i));
7          range = max(inputMlp(:,i)) - mi;
8          temp = ((inputMlp(:,i)-mi)*(1+1)/(range))-1;
9          inputMlpN = [inputMlpN temp];
10     end
11 end
```

Kode Sumber 4. 8 Kode program fungsi normalz()

Setelah setiap fitur dinormalisasi selanjutnya disimpan dalam sebuah variabel inputMlpN.

4.2.2.2.3 Fungsi feedForward()

Fungsi ini digunakan untuk melakukan proses *feedforward* pada *Multilayer Perceptron*. Kode program ditunjukkan pada Kode Sumber 4.9.

```
1  function [outputInputZj, outputHiddenYk] =
   feedForward(hiddenLayer,inputRow,inputWeight,hiddenWei
   ght,bias1,bias2)
```

```
2
3       %hitung layer input ke hidden
4       outputInputZnetj = [];
5       outputInputZj = [];
6       for i=1:hiddenLayer
7           temp = bias1(1,i) +
    sum(inputWeight(i,:).*inputRow);
8           outputInputZnetj = [outputInputZnetj temp];
9           temp2 = 1/(1 + exp(-1*temp));
10          outputInputZj = [outputInputZj temp2];
11      end
12
13      %hitung layer hidden ke output
14      outputHiddenYnetk = [];
15      outputHiddenYk = [];
16      for i=1:3
17          temp = bias2(1,i) +
18  sum(hiddenWeight(i,:).*outputInputZj);
19          outputHiddenYnetk  = [outputHiddenYnetk temp];
20          temp2 = 1/(1 + exp(-1*temp));
21          outputHiddenYk = [outputHiddenYk temp2];
22      end
23  end
```

Kode Sumber 4. 9 Kode program fungsi feedForward()

Variabel *outputInputZj* adalah hasil kalkulasi antara masukkan dan bobot pada lapisan masukkan dan lapisan tersembunyi. Sedangkan variabel *outputHiddenYk* adalah hasil kalkulasi antara *outputInputZj* dan bobot pada lapisan tersembunyi dan lapisan keluaran. Kedua variabel ini digunakan sebagai masukkan proses *backpropagatiaon*.

4.2.2.2.4 Fungsi backPropagation()

Fungsi ini digunakan untuk melakukan proses *backpropagation*. *Backpropagation* adalah proses perhitungan eror dari bobot dan bias. Proses ini dilakukan berulang-ulang sejumlah data pembelajaran dan *epoch* sehingga menghasilkan model atau bobot dan bias yang terbaik. Kode Sumber 4.10 menunjukkan kode program dari fungsi ini.

```
function [deltaInput, deltaHiddenWjk, deltabias1,
deltabias2] =
backPropagation(learningRate,inputRow,outputInputZj,ou
tputHiddenYk,outputRow,hiddenWeight,bias1,bias2)
    %layer output ke hidden
    %menghitung informasi error  dk
    d_k = (outputRow -
outputHiddenYk).*outputHiddenYk.*(1 - outputHiddenYk);

    %menghitung delta layer hidden ke output
    deltaHiddenWjk = [];
    for i=1:size(d_k,2)
        temp = learningRate*d_k(i).*outputInputZj;
        deltaHiddenWjk = [deltaHiddenWjk; temp];
    end
    %delta bias2
    deltabias2 = learningRate.*d_k.*bias2;

    %layer hidden ke input
    %menghitung koreksi error d_inj
    d_inj = [];
    for i=1:size(hiddenWeight,2)
        temp =  d_k*hiddenWeight(:,i);
        d_inj = [d_inj temp];
    end
    dj = d_inj.*outputInputZj.*(1 - outputInputZj);

    %menghitung delta layer input ke hidden dan bias1
    deltaInput = [];
    for i=1:size(d_inj,2)
        temp = learningRate*dj(i)*inputRow;
        deltaInput = [deltaInput; temp];
    end

    %delta bias1
    deltabias1 = learningRate.*dj.*bias1;

end
```

Kode Sumber 4. 10 Kode program fungsi backPropagation()

d_k adalah eror antara hasil pembelajaran pada *feedforward* dan target. Selanjutnya nilai *d_k* digunakan untuk menghitung selisih bobot dan bias lapisan keluaran. *d_inj* dan *dj* adalah eror pada keluaran lapisan tersembunyi. Nilai tersebut digunakan untuk menghitung selisih bobot dan bias lapisan

tersembunyi. Keluaran dari fungsi ini yaitu deltaInput, deltaHiddenWjk, deltabias1, dan deltabias2.

4.2.2.2.5 Fungsi updateWeightBias()

Fungsi ini hanya digunakan untuk melakukan pembaruan pada bobot dan bias setelah melalui proses *feedforward* dan *backpropagation*. Keluaran dari fungsi ini adalah bobot dan bias yang lebih baik dari sebelumnya. Kode Sumber 4.11 menunjukkan kode program pembaruan bobot dan bias.

```
1  function [inputWeightNew, hiddenWeightNew, newBias1,
   newBias2] = updateWeightBias(inputWeight,hiddenWeight,
   bias1, bias2, deltaInput, deltaHiddenWjk, deltabias1,
   deltabias2)
2      inputWeightNew = inputWeight + deltaInput;
3      hiddenWeightNew = hiddenWeight + deltaHiddenWjk;
4      newBias1 = bias1 + deltabias1;
5      newBias2 = bias2 + deltabias2;
6  end
```

Kode Sumber 4. 11 Kode program fungsi updateWeightBias()

4.2.2.2.6 Fungsi mainTestMlp()

Fungsi ini adalah fungsi yang digunakan untuk melakukan pengujian terhadap data uji menggunakan model yang telah dihasilkan dari proses sebelumnya. Data uji satu per satu dibaca kemudian menghitung keluaran melalui sebuah fungsi lain yaitu testMlp(). Keluaran akhir fungsi ini adalah sebuah matriks hasil kalkulasi seluruh data uji pada model, resultMlp. Kode Sumber 4.12 menunjukkan kode program fungsi mainTestMlp().

```
function [resultMlp] = mainTestMlp(hiddenLayer,
inputTestN, inputWeight, hiddenWeight, bias1, bias2)

    resultMlp = [];
    for i=1:size(inputTestN,1)
        inputTestRow = inputTestN(i,:);
        [res] = testMlp(hiddenLayer, inputTestRow,
inputWeight, hiddenWeight, bias1, bias2);
        resultMlp = [resultMlp;res];
    end
end
```

Kode Sumber 4. 12 Kode program fungsi mainTestMlp()

4.2.2.2.7 Fungsi testMlp()

Fungsi ini adalah fungsi untuk menghitung keluaran dari sebuah data uji. Keluaran ini sejumlah 3 nilai sesuai dengan jumlah node keluaran. Kode Sumber 4.13 menunjukkan fungsi testMlp().

```
function [result] = testMlp(hiddenLayer, inputTestRow,
inputWeight, hiddenWeight, bias1, bias2)
    output1 = [];
    for i=1:hiddenLayer
        temp = bias1(i) +
sum(inputWeight(i,:).*inputTestRow);
        temp = 1/(1 + exp(-1*temp));
        output1 = [output1, temp];
    end
    result = [];
    for i=1:3
        temp = bias2(i) +
sum(hiddenWeight(i,:).*output1);
        output2 = 1/(1 + exp(-1*temp));
        result = [result output2];
    end
end
```

Kode Sumber 4. 13 Kode program fungsi testMlp()

Kode program dari fungsi ini sama seperti proses *feedforward* pada tahap pembelajaran. Hasil dari fungsi ini adalah 3 nilai keluaran yang nantinya digunakan untuk menentukan kelas setiap data ujinya.

4.2.2.2.8 Fungsi validate()

Fungsi ini adalah fungsi yang digunakan untuk mengukur performa dari model dengan menghitung *accuracy*, *sensitivity*, dan *specificity* dari hasil klasifikasi. Kode Sumber 4.14 menunjukkan kode program fungsi validate().

```
function [acc, spec, sens, confusionM] = validate(resultMlp, outputTest)
    spec = zeros(1,3);
    sens = zeros(1,3);
    confusionM = zeros(3,3);
    for i=1:size(resultMlp,1)
       [M1, I1] = max(resultMlp(i,:));
       [M2, I2] = max(outputTest(i,:));
       confusionM(I2, I1) = confusionM(I2, I1)+1;
    end

    %accuracy
    acc = (confusionM(1,1)+confusionM(2,2)+confusionM(3,3))/sum(sum(confusionM));

    %sensitivity
    for i=1:3
        temp = confusionM(i,i)/sum(confusionM(i,:));
        sens(1,i) = temp;
    end

    %specificity
    tn = (confusionM(2,2)+confusionM(2,3)+confusionM(3,2)+confusionM(3,3));
    spec(1,1) = tn/(tn+confusionM(2,1)+confusionM(3,1));
    tn = (confusionM(1,1)+confusionM(1,3)+confusionM(3,1)+confusionM(3,3));
    spec(1,2) = tn/(tn+confusionM(1,2)+confusionM(3,2));
    tn = (confusionM(1,1)+confusionM(1,2)+confusionM(2,1)+confusionM(2,2));

    spec(1,3) = tn/(tn+confusionM(1,3)+confusionM(2,3));
end
```

Kode Sumber 4. 14 Kode program fungsi validate()

Karena jumlah kelas dari data adalah 3 sehingga *confusion matrix* yang dibuat berukuran 3x3 seperti pada baris 4. Baris 5 hingga 9 adalah proses penentuan dari kelas tiap data uji. Dari 3 nilai keluaran, pilih nilai yang tertinggi dan diambil indeksnya sebagai kelas. Baris 12 adalah penghitungan *accuracy*. Baris 15 hingga 18 adalah perhitungan *sensitivity* sedangkan baris 21 hingga 26 adalah *specificity*. Setiap kelas memiliki nilai *sensitivity* dan *specificity* nya masing-masing.

BAB V
UJI COBA DAN EVALUASI

Pada bab kelima ini akan dijelaskan mengenai skenario dan uji coba perangkat lunak yang telah dibangun. Selain itu, hasil uji coba akan dievaluasi kinerjanya sehingga dapat diputuskan apakah perangkat lunak ini mampu menyelesaikan permasalahan yang telah dirumuskan di awal.

5.1 Lingkungan Uji Coba

Sebelumnya, perlu diketahui lingkungan uji coba, baik perangkat keras maupun perangkat lunak, yang digunakan pada uji coba Tugas Akhir ini. Lingkungan tersebut ditunjukkan pada Tabel 5.1 berikut ini.

Tabel 5. 1 Spesifikasi lingkungan uji coba

Perangkat	Spesifikasi
Perangkat keras	Prosesor: Intel® Pentium® 2020M 2.40GHz Memori: 6.00 GB
Perangkat lunak	Sistem Operasi: Microsoft Windows 8.1 64-bit Perangkat Pengembang: Matlab R2013a Perangkat Pembantu: Microsoft Excel 2013

5.2 Data Uji Coba

Seperti yang telah dijelaskan sebelumnya, data yang digunakan adalah data sinyal otak manusia (EEG) yang diunduh dari website milik "Klinik für Epileptologie, Universität Bonn". Data ini terdiri dari 5 set rekaman EEG (A-E) dan masing-masing set dipilih 20 file secara acak. Daftar nama-nama file tersebut ditunjukkan pada Tabel 5.2.

Tabel 5. 2 File yang digunakan sebagai masukan perangkat lunak

A	B	C	D	E
Z006	O003	N004	F006	S003
Z008	O006	N005	F009	S025
Z012	O008	N014	F014	S027
Z016	O013	N021	F016	S028
Z019	O017	N033	F018	S030
Z023	O030	N040	F023	S037
Z027	O038	N049	F026	S038
Z030	O042	N051	F029	S039
Z031	O047	N054	F032	S040
Z033	O051	N063	F034	S045
Z043	O062	N067	F035	S063
Z046	O064	N068	F039	S070
Z050	O067	N071	F048	S075
Z051	O069	N082	F058	S077
Z056	O073	N084	F069	S082
Z063	O076	N087	F073	S083
Z088	O080	N091	F086	S086
Z091	O083	N092	F087	S091
Z098	O085	N098	F098	S092
Z100	O093	N100	F099	S099

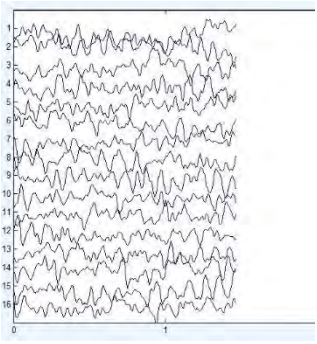
Data mentah diolah dengan ICA dan Wavelet hingga menghasilkan 320 data untuk masing-masing kelas sehingga total terdapat 1600 data (20 file dengan 16x20 sebanyak 5 data set). Untuk uji coba, metode yang digunakan adalah *K-Fold Cross Validation* dengan $K = 10$. Pada MLP, jumlah lapisan tersembunyi sebanyak 15 node. *Learning rate* yang digunakan 0.1 dan *epoch* sebanyak 200. Data diacak kemudian dibagi menjadi 10 bagian.

Secara bergantian semua bagian menjadi data pengujian dan pembelajaran pada proses klasifikasi seperti yang telah dijelaskan pada bab rancangan perangkat lunak.

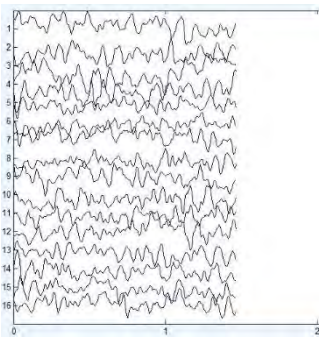
5.3. *Preprocessing* Data

Tahap pertama adalah melakukan prepsocessing dengan SCICA dan WICA. Pada SCICA, setiap satu file rekaman EEG dibagi menjadi 16 bagian selanjutnya dihilangkan derau-nya dengan EEGLAB. Gambar 5.1 hingga 5.5 adalah contoh visualisasi hasil SCICA setiap data set. Gambar sebelah kiri sebelum dan sebelah kanan setelah ICA.

Sedangkan WICA, setiap satu file rekaman EEG didekomposisi dengan SWT menjadi 5 *subband*. Selanjutnya dihilangkan deraunya dengan EEGLAB dan dijadikan menjadi satu sinyal utuh dengan proses rekonstruksi. Gambar 5.6 hingga 5.10 adalah contoh visualisasi hasil WICA setiap data set. Gambar sebelah atas sebelum dan sebelah bawah setelah ICA.

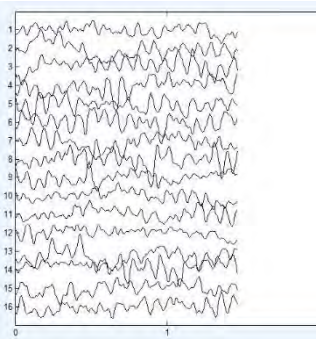


(a). Sebelum ICA

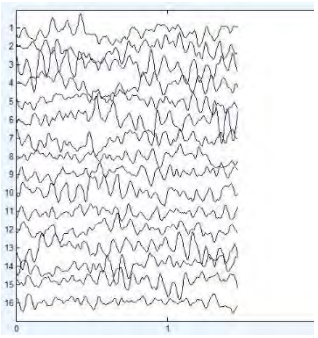


(b). Setelah ICA

Gambar 5. 1 Hasil SCICA sampel data set A, Z006.txt

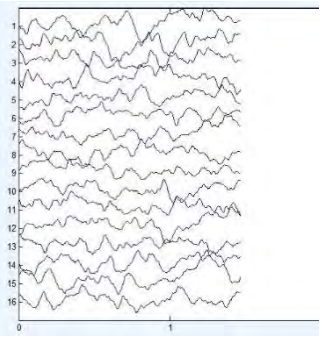


(a). Sebelum ICA

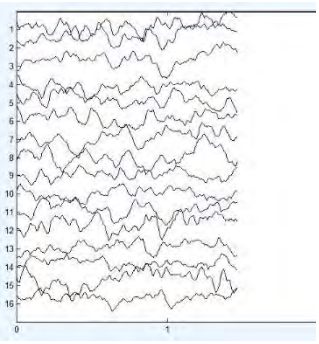


(b). Setelah ICA

Gambar 5. 2 Hasil SCICA sampel data set B, O003.txt

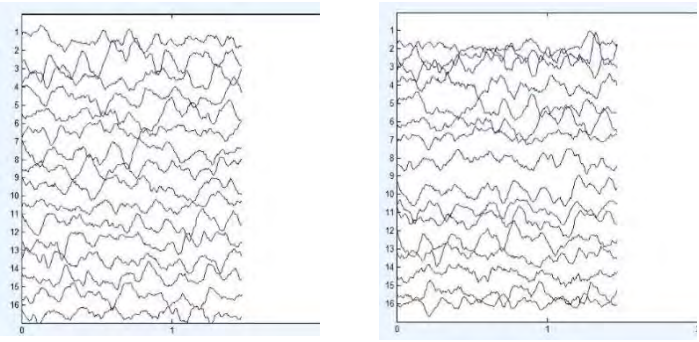


(a). Sebelum ICA



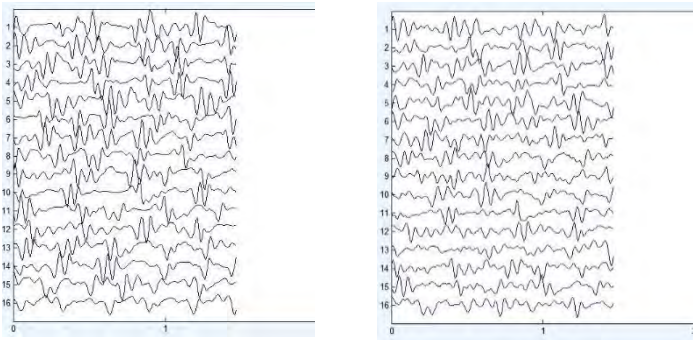
(b). Setelah ICA

Gambar 5. 3 Hasil SCICA sampel data set C, N004.txt



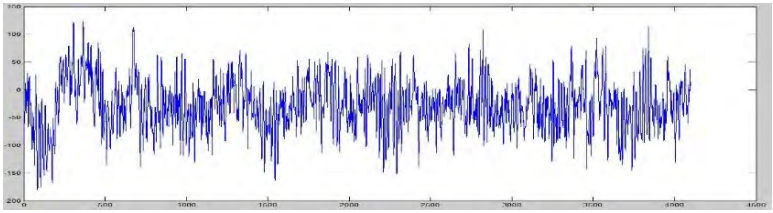
(a). Sebelum ICA (b). Setelah ICA

Gambar 5. 4 Hasil SCICA sampel data set D, F006.txt

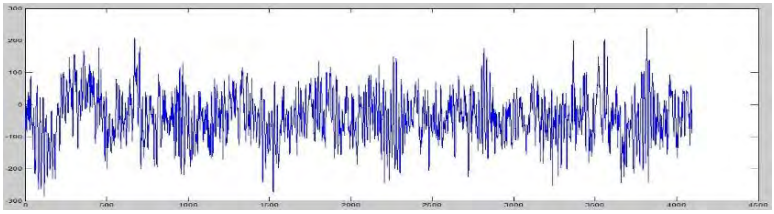


(a). Sebelum ICA (b). Setelah ICA

Gambar 5. 5 Hasil SCICA sampel data set E, S003.txt

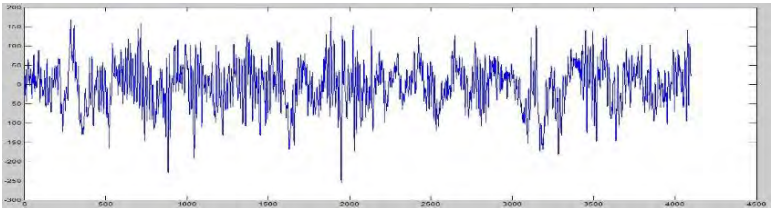


(a). Sebelum ICA

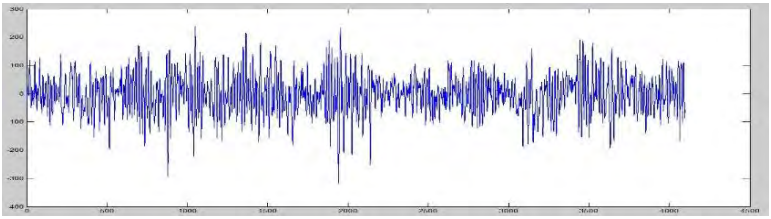


(b). Setelah ICA

Gambar 5. 6 Hasil WICA sampel data set A, Z006.txt

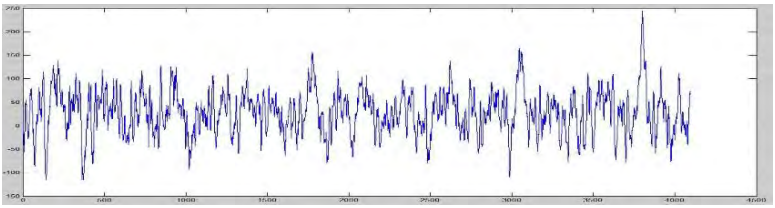


(a). Sebelum ICA

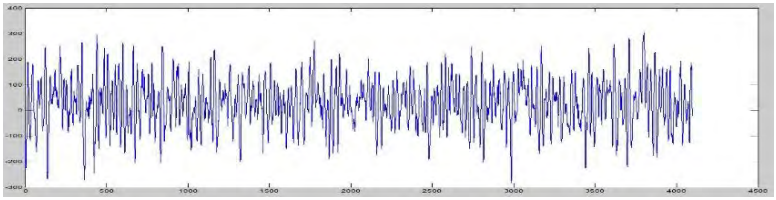


(b). Setelah ICA

Gambar 5. 7 Hasil WICA sampel data set B, O003.txt

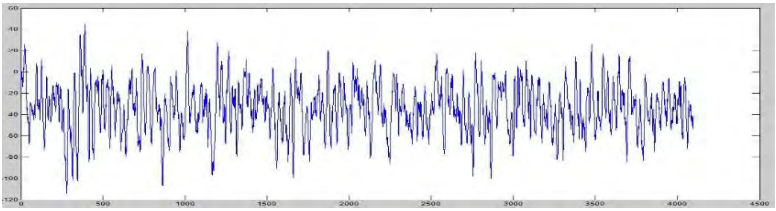


(a). Sebelum ICA

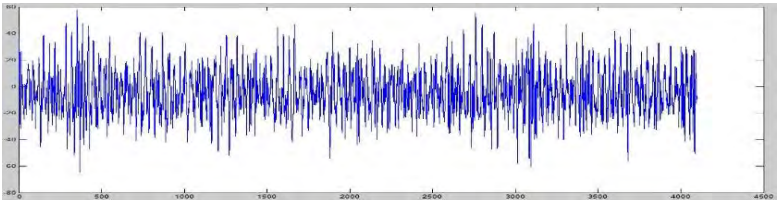


(b). Setelah ICA

Gambar 5. 8 Hasil WICA sampel data set C, N004.txt

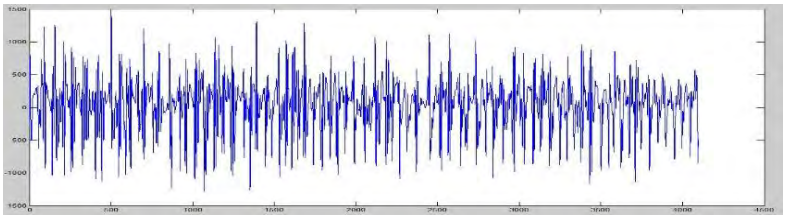


(a). Sebelum ICA

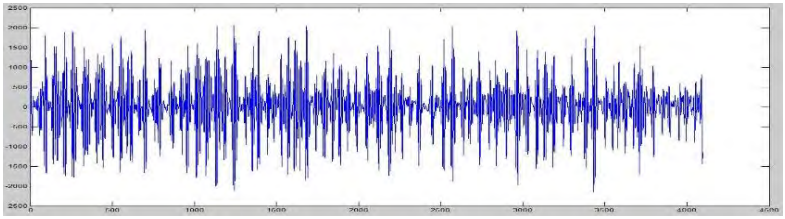


(b). Setelah ICA

Gambar 5. 9 Hasil WICA sampel data set D, F006.txt



(a). Sebelum ICA



(b). Setelah ICA

Gambar 5. 10 Hasil WICA sampel data set E, S003.txt

5.4. Skenario Uji Coba

Sebelum melakukan uji coba, perlu ditentukan skenario yang akan digunakan dalam uji coba. Melalui skenario ini, perangkat lunak diuji apakah sudah berjalan dengan benar dan memiliki performa yang baik sesuai dengan kondisi yang ditentukan. Selain itu juga membandingkan antara kedua *preprocessing,* SCICA dan WICA, manakah yang memiliki hasil yang lebih baik dengan skenario yang berbeda. Terdapat 2 macam skenario uji coba, yaitu :

1. Perhitungan performa dengan mengubah level DWT pada proses ekstraksi fitur sedangkan *mother wavelet*-nya tetap yaitu Daubechies 2. Ada 4 macam level yang diuji yaitu 4, 5, 6, dan 7.
2. Perhitungan performa dengan mengubah jenis Daubechies pada proses ekstraksi fitur. Ada 4 jenis Daubechies yang diuji, yaitu db2, db4, db6, dan db8 sedangkan levelnya tetap yaitu 4.

5.4.1. Skenario Uji Coba 1

Skenario uji coba 1 adalah perhitungan *accuracy*, *sensitivity*, dan *specificity* dengan mencoba berbagai level DWT pada proses ekstraksi fitur yaitu 4, 5, 6, dan 7 Daubechies 2. Setiap level menghasilkan jumlah fitur yang berbeda-beda seperti yang ditunjukkan pada Tabel 5.3. DWT dengan 4 level menghasilkan 20 fitur, 5 level menghasilkan 24 fitur, 6 level menghasilkan 28 fitur, dan 7 level menghasilkan 32 fitur.

Selanjutnya setiap level diuji dengan MLP 10-*Fold Cross Validation* masing-masing sebanyak 5 kali. Berdasarkan hasil tersebut dilakukan penghitungan performa dengan membuat *Confusion Matrix* dan rumus-rumus yang ada pada Persamaan 2.8, 2.9, dan 2.10. Tabel 5.3 dan 5.4 berikut ini adalah hasil rata-rata penghitungan performa tiap level dari SCICA dan WICA.

Sebagai keterangan, *sensitivity* dan *specificity* masing-masing berjumlah 3. Angka 1 untuk menunjukkan kelas 1, yaitu normal, angka 2 menunjukkan kelas 2, yaitu epilepsi tidak kejang, dan angka 3 menunjukkan kelas 3, yaitu epilepsi kejang.

Tabel 5. 3 Rata-rata performa SCICA setiap level DWT

Level	***Accuracy***	***Sensitivity***			***Specificity***		
		1	**2**	**3**	**1**	**2**	**3**
4	89.67	94.6	89.56	80.39	90.74	93.63	98.86
5	89.26	96.37	81.95	89.97	89.68	96.27	97.15
6	89.35	88.15	88.24	92.94	95.91	92.86	95.06
7	**90.31**	**93.8**	**85.9**	**91.72**	**93.62**	**95.31**	**96.18**
Rata-rata	89.65	93.23	86.41	88.75	92.49	94.52	96.81

Tabel 5. 4 Rata-rata performa WICA setiap level DWT

Level	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
4	74.58	93.82	54.69	75.11	69.66	91.36	97.5
5	**79.9**	**77.82**	**83.85**	**76.06**	**89.13**	**79.51**	**98.43**
6	76.7	69.01	86.14	72.97	90.76	72.53	98.51
7	76.8	70.12	83.28	77.36	89.51	74.67	97.85
Rata-rata	76.99	77.69	76.99	75.37	84.76	79.52	98.07

Secara umum hasil yang ditunjukkan oleh penghitungan performa di atas, SCICA memiliki hasil klasifikasi yang jauh lebih baik dibandingkan dengan WICA pada semua level uji coba. Hal ini ditunjukkan dengan rata-rata akurasi SCICA sebesar 89.65% sedangkan WICA 76.99% saja. Selain itu dari segi waktu komputasi rata-rata SCICA juga lebih cepat, yaitu 16.96 menit sedangkan WICA 18.78 menit.

Pada SCICA, dari 4 jenis level yang diujicobakan, yang terbaik adalah ekstraksi fitur DWT Daubechies 2 dengan 7 level dengan rata-rata *accuracy* 90.31%, *sensitivity* kelas 1 yaitu 93.23%, kelas 2 yaitu 85.9%, kelas 3 yaitu 91.72%, dan *specificity* kelas 1 yaitu 93.62%, kelas 2 yaitu 95.31%, kelas 3 yaitu 96.18%. Namun waktu komputasi yang dibutuhkan lebih lama dibandingkan level lain yaitu rata-rata 17.29 menit karena jumlah fiturnya yang memang lebih banyak sehingga membutuhkan waktu yang lebih lama.

Pada WICA, dari 4 jenis level yang diujicobakan, yang terbaik adalah ekstraksi fitur DWT Daubechies 2 dengan 5 level dengan rata-rata *accuracy* 79.9%, *sensitivity* kelas 1 yaitu 77.82%, kelas 2 yaitu 83.85%, kelas 3 yaitu 76.06%, dan *specificity* kelas 1 yaitu 89.13%, kelas 2 yaitu 79.51%, kelas 3 yaitu 98.43%. Waktu

komputasi yang dibutuhkan yaitu rata-rata 18.09 menit. Hasil uji coba 1 lebih lengkap terdapat di lampiran.

5.4.2. Skenario Uji Coba 2

Skenario uji coba 1 adalah perhitungan *accuracy*, *sensitivity*, dan *specificity* dengan mencoba berbagai jenis *mother wavelet* Daubechies DWT pada proses ekstraksi fitur yaitu db2, db4, db6, dan db8 dengan 4 level. Setiap jenis Daubechies menghasilkan jumlah fitur tetap yaitu 20 fitur namun dengan nilai yang berbeda-beda. Selanjutnya setiap data Daubechies diuji dengan MLP 10-*Fold Cross Validation* masing-masing sebanyak 5 kali. Berdasarkan hasil tersebut dilakukan penghitungan performa dengan membuat *Confusion Matrix* dan rumus-rumus yag ada pada Persamaan 2.8, 2.9, dan 2.10. Tabel 5.5 dan 5.6 berikut ini adalah hasil rata-rata penghitungan performa tiap jenis Daubechies dari SCICA dan WICA.

Tabel 5. 5 Rata-rata performa SCICA setiap jenis Daubechies DWT

Db	***Accuracy***	***Sensitivity***			***Specificity***		
		1	**2**	**3**	**1**	**2**	**3**
2	89.68	94.6	89.56	80.39	90.74	93.63	98.86
4	87.3	79.42	96.65	84.15	96.3	84.27	98.67
6	**92.09**	**91.15**	**92.18**	**93.43**	**95.73**	**95.37**	**96.79**
8	91.25	90.59	94.41	86.34	94.97	92.8	98.25
Rata-rata	90.08	88.94	93.2	86.08	94.44	91.52	98.14

Sama seperti uji coba 1, secara umum hasil yang ditunjukkan oleh penghitungan performa di atas, SCICA memiliki hasil klasifikasi yang jauh lebih baik dibandingkan dengan WICA pada semua level uji coba. Hal ini ditunjukkan dengan rata-rata

SCICA sebesar 90.08% sedangkan WICA 79.65% saja. Selain itu dari segi waktu komputasi rata-rata SCICA juga lebih cepat, yaitu 18.06 menit sedangkan WICA 18.48 menit.

Tabel 5. 6 Rata-rata performa WICA setiap jenis Daubechies DWT

Db	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
2	74.58	93.82	54.69	75.11	69.66	91.36	97.5
4	80.21	71.29	90.18	78.59	92.39	76.15	98.83
6	**82.4**	**85.25**	**77.83**	**86.29**	**85.38**	**89.83**	**96.61**
8	81.42	90.06	70.44	86	78.78	92.36	98.43
Rata-rata	79.65	85.1	73.28	81.5	81.55	87.43	97.84

Pada SCICA, dari 4 jenis Daubechies yang diujicobakan, yang terbaik adalah ekstraksi fitur DWT Daubechies 6 dengan 4 level dengan rata-rata *accuracy* 92.08%, *sensitivity* kelas 1 yaitu 91.15%, kelas 2 yaitu 92.18%, kelas 3 yaitu 93.43%, dan *specificity* kelas 1 yaitu 95.37%, kelas 2 yaitu 96.79%, kelas 3 yaitu 93.82% serta waktu komputasi rata-rata yang dibutuhkan 17.24.

Pada WICA, dari 4 jenis Daubechies yang diujicobakan, yang terbaik adalah ekstraksi fitur DWT Daubechies 6 dengan 4 level dengan rata-rata *accuracy* 82.4%, *sensitivity* kelas 1 yaitu 85.25%, kelas 2 yaitu 77.83%, kelas 3 yaitu 86.29%, dan *specificity* kelas 1 yaitu 85.38%, kelas 2 yaitu 89.83%, kelas 3 yaitu 96.61% serta waktu komputasi yang dibutuhkan rata-rata 18.56 menit. Hasil uji coba 2 lebih lengkap terdapat di lampiran.

5.5. Evaluasi Umum Skenario Uji Coba

Berdasarkan kedua uji coba skenario di atas, diketahui bahwa *preprocessing* SCICA memiliki nilai yang lebih baik dibandingkan dengan *preprocessing* WICA. Skenario pertama,

SCICA dengan ekstraksi fitur DWT Daubechies 2, jumlah level yang terbaik saat mendekomposisi yaitu 7 dengan akurasi sebesar 90.31%. Skenario kedua, SCICA dengan ekstraksi fitur DWT 4 level, jenis *mother wavelet* yang terbaik adalah Daubechies 6 dengan akurasi sebesar 92.09%. Kedua skenario berjalan dengan rata-rata waktu komputasi yang sama yaitu 17 menit. Bila dibandingkan, akurasi terbaik yaitu pada skenario kedua. Namun, *sensitivity* kelas 1 pada skenario pertama lebih baik dibandingkan dengan skenario kedua, yaitu dengan selisih 2.65%. Hal ini disebabkan karena jumlah *true positive*-nya yang lebih tinggi atau jumlah *e12* dan *e13* lebih rendah. Hasil perbandingan keduanya ditunjukkan pada Tabel 5.7 berikut ini.

Tabel 5. 7 Perbandingan hasil uji coba setiap skenario

Skenario Uji Coba	***Accuracy***	***Sensitivity***			***Specificity***		
		1	**2**	**3**	**1**	**2**	**3**
1	90.31	93.8	85.9	91.72	93.62	95.31	96.18
2	**92.09**	**91.15**	**92.18**	**93.43**	**95.73**	**95.37**	**96.79**

[Halaman ini sengaja dikosongkan]

BAB VI
KESIMPULAN DAN SARAN

Bab VI ini membahas tentang kesimpulan yang didasari oleh hasil uji coba pada bab sebelumnya. Kesimpulan tersebut nantinya menjawab rumusan masalah yang telah ada pada pendahuluan. Selain itu, juga terdapat saran sebagai acuan untuk mengembangkan topik Tugas Akhir ini lebih lanjut di masa depan.

6.3. Kesimpulan

Dari hasil uji coba yang telah dilakukan, dapat diambil kesimpulan sebagai berikut:

1. Implementasi metode *Independent Component Analysis, Wavelet Transform,* dan *Multilayer Perceptron* mampu mendeteksi penyakit epilepsi sehingga dapat dijadikan metode dalam perangkat lunak pendeteksi epilepsi.
2. Metode penghilangan derau *Single Channel Independent Component Analysis* (SCICA) pada *Independent Component Analysis* menghasilkan akurasi yang lebih baik bila dibandingkan dengan *Wavelet Independent Component Analysis* (WICA) dalam *preprocessing* data
3. Pada tahap ekstraksi fitur dengan *Wavelet Transform,* tingkat akurasi yang terbaik pada klasifikasi adalah dengan menggunakan *mother wavelet* Daubechies 6 dengan 4 level, yaitu 92.09%.

6.4. Saran

Saran yang diberikan untuk pengembangan perangkat lunak ini adalah:

1. Guna mengurangi waktu komputasi, perlu dilakukan pereduksian fitur, misalnya dengan metode *Principal Component Analysis* (PCA).
2. Jika diperlukan, perangkat lunak bisa dilengkapi dengan GUI sehingga lebih mudah dalam penggunaan.

[Halaman ini sengaja dikosongkan]

BAB VI
KESIMPULAN DAN SARAN

Bab VI ini membahas tentang kesimpulan yang didasari oleh hasil uji coba pada bab sebelumnya. Kesimpulan tersebut nantinya menjawab rumusan masalah yang telah ada pada pendahuluan. Selain itu, juga terdapat saran sebagai acuan untuk mengembangkan topik Tugas Akhir ini lebih lanjut di masa depan.

6.3. Kesimpulan

Dari hasil uji coba yang telah dilakukan, dapat diambil kesimpulan sebagai berikut:

1. Implementasi metode *Independent Component Analysis, Wavelet Transform,* dan *Multilayer Perceptron* mampu mendeteksi penyakit epilepsi sehingga dapat dijadikan metode dalam perangkat lunak pendeteksi epilepsi.
2. Metode penghilangan derau *Single Channel Independent Component Analysis* (SCICA) pada *Independent Component Analysis* menghasilkan akurasi yang lebih baik bila dibandingkan dengan *Wavelet Independent Component Analysis* (WICA) dalam *preprocessing* data
3. Pada tahap ekstraksi fitur dengan *Wavelet Transform,* tingkat akurasi yang terbaik pada klasifikasi adalah dengan menggunakan *mother wavelet* Daubechies 6 dengan 4 level, yaitu 92.09%.

6.4. Saran

Saran yang diberikan untuk pengembangan perangkat lunak ini adalah:

1. Guna mengurangi waktu komputasi, perlu dilakukan pereduksian fitur, misalnya dengan metode *Principal Component Analysis* (PCA).
2. Jika diperlukan, perangkat lunak bisa dilengkapi dengan GUI sehingga lebih mudah dalam penggunaan.

[Halaman ini sengaja dikosongkan]

LAMPIRAN

Tabel A. 1 *Confussion matrix* uji coba skenario 1 SCICA level 4, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	65	2	0
	2	4	58	1
	3	3	3	24

Tabel A. 2 *Confussion matrix* uji coba skenario 1 SCICA level 5, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	69	0	4
	2	11	50	0
	3	1	1	24

Tabel A. 3 *Confussion matrix* uji coba skenario 1 SCICA level 6, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	41	10	2
	2	4	60	5
	3	1	1	36

Tabel A. 4 *Confussion matrix* uji coba skenario 1 SCICA level 7, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	64	2	2
	2	7	59	3
	3	0	1	22

Tabel A. 5 *Confussion matrix* uji coba skenario 1 WICA level 4, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	52	7	0
	2	26	35	3
	3	2	5	30

Tabel A. 6 *Confussion matrix* uji coba skenario 1 WICA level 5, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	51	15	0
	2	8	52	2
	3	0	7	25

Tabel A. 7 *Confussion matrix* uji coba skenario 1 WICA level 6, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	41	15	0
	2	4	59	0
	3	0	7	34

Tabel A. 8 *Confussion matrix* uji coba skenario 1 WICA level 7, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	38	20	0
	2	11	54	4
	3	1	7	25

Tabel A. 9 *Confussion matrix* uji coba skenario 2 SCICA Daubechies 2, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	65	2	0
	2	4	58	1
	3	3	3	24

Tabel A. 10 *Confussion matrix* uji coba skenario 2 SCICA Daubechies 4, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	44	17	2
	2	2	63	0
	3	4	2	26

Tabel A. 11 *Confussion matrix* uji coba skenario 2 SCICA Daubechies 6, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	51	5	1
	2	2	63	1
	3	1	0	36

Tabel A. 12 *Confussion matrix* uji coba skenario 2 SCICA Daubechies 8, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	54	4	1
	2	2	65	0
	3	2	0	32

Tabel A. 13 *Confussion matrix* uji coba skenario 2 WICA Daubechies 2, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	52	7	0
	2	26	35	3
	3	2	5	30

Tabel A. 14 *Confussion matrix* uji coba skenario 2 WICA Daubechies 4, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	50	14	0
	2	8	42	6
	3	4	6	30

Tabel A. 15 *Confussion matrix* uji coba skenario 2 WICA Daubechies 6, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	61	7	1
	2	9	48	2
	3	1	3	28

Tabel A. 16 *Confussion matrix* uji coba skenario 2 WICA Daubechies 8, 10 Fold Cross Validation, Fold ke-1

		Prediksi		
		1	**2**	**3**
Aktual	**1**	54	8	0
	2	12	52	3
	3	3	2	26

Tabel A. 17 Rekap performa hasil uji coba skenario 1 SCICA level 4

Percobaan	***Accuracy***	***Sensitivity***			***Specificity***		
		1	**2**	**3**	**1**	**2**	**3**
1	89.18	94.08	89.22	79.718	90.48	93.24	98.75
2	89.93	94.97	89.22	81.607	91.21	93.96	98.59
3	89.86	94.97	89.36	81.134	90.54	93.84	99.06
4	89.86	94.97	89.36	81.134	90.54	93.84	99.06
5	89.62	94.97	89.85	78.811	90.28	93.76	99.06
Rata-rata	**89.7**	**94.79**	**89.4**	**80.481**	**90.61**	**93.73**	**98.91**

Tabel A. 18 Rekap performa hasil uji coba skenario 1 SCICA level 5

Percobaan	***Accuracy***	***Sensitivity***			***Specificity***		
		1	**2**	**3**	**1**	**2**	**3**
1	89.56	96.30	82.4	90.75	89.78	96.45	97.32
2	89.12	96.31	81.61	90.04	89.57	96.25	97.09
3	89.12	96.31	81.61	90.04	89.57	96.25	97.09
4	89.44	96.60	82.39	89.44	89.96	96.24	97.16
5	89.06	96.31	81.77	89.6	89.54	96.14	97.08
Rata-rata	**89.26**	**96.36**	**81.95**	**89.97**	**89.68**	**96.27**	**97.15**

Tabel A. 19 Rekap performa hasil uji coba skenario 1 SCICA level 6

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	89.12	87.85	88.02	92.94	95.81	92.86	94.89
2	89.68	88.98	88.18	93.2	95.82	93.45	95.12
3	89.44	88.29	88.35	92.85	96.02	92.63	95.27
4	89.25	87.99	88.17	92.7	96	92.83	94.87
5	89.25	87.63	88.5	93	95.91	92.52	95.17
Rata-rata	**89.35**	**88.15**	**88.24**	**92.94**	**95.91**	**92.86**	**95.06**

Tabel A. 20 Rekap performa hasil uji coba skenario 1 SCICA level 7

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	90.5	93.9	86.29	91.75	92.99	95.6	96.66
2	90.12	93.56	85.81	91.41	93.62	94.96	96.19
3	90.12	93.43	85.12	92.92	93.61	95.6	95.74
4	90.18	94.05	85.82	90.9	93.73	95.29	95.98
5	90.62	94.04	86.44	91.6	94.13	95.09	96.34
Rata-rata	**90.31**	**93.8**	**85.9**	**91.72**	**93.62**	**95.31**	**96.18**

Tabel A. 21 Rekap performa hasil uji coba skenario 1 WICA level 4

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	74.81	94.1	54.64	75.61	69.69	91.45	97.68
2	74.5	93.4	54.82	75.2	69.61	91.56	97.35
3	74.5	93.95	54.9	74.44	69.66	91.36	97.43
4	74.25	93.55	54.43	74.67	69.66	90.98	97.35
5	74.81	94.1	54.64	75.61	69.69	91.45	97.68
Rata-rata	**74.58**	**93.82**	**54.69**	**75.11**	**69.66**	**91.36**	**97.5**

Tabel A. 22 Rekap performa hasil uji coba skenario 1 WICA level 5

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	79.56	77.57	83.53	75.39	88.69	79.15	98.6
2	80.31	78.05	84.54	76.41	89.55	79.85	98.36
3	79.88	77.88	83.36	76.81	88.79	79.76	98.44
4	80.06	77.99	84.35	75.48	89.45	79.52	98.36
5	79.68	77.6	83.47	76.22	89.16	79.28	98.37
Rata-rata	**79.9**	**77.82**	**83.85**	**76.06**	**89.13**	**79.51**	**98.43**

Tabel A. 23 Rekap performa hasil uji coba skenario 1 WICA level 6

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	76.88	70.06	85.13	73.94	90.31	73.68	98.24
2	76.5	68.09	86.5	72.92	91.01	71.89	98.53
3	76.12	68.49	85.84	72.37	90.62	71.82	98.4
4	77.62	70.11	86.44	74.5	91.02	73.64	98.61
5	76.38	68.29	86.8	71.12	90.81	71.59	98.77
Rata-rata	**76.7**	**69.01**	**86.14**	**72.97**	**90.76**	**72.53**	**98.51**

Tabel A. 24 Rekap performa hasil uji coba skenario 1 WICA level 7

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	77.18	70.63	83.92	76.83	89.36	74.86	98.27
2	76.31	68.61	83.36	77.98	90.24	73.73	97.41
3	77	72.15	81.82	77.17	88.36	76.14	97.89
4	77.18	70.63	83.92	76.83	89.36	74.86	98.27
5	76.31	68.61	83.36	77.98	90.24	73.73	97.41
Rata-rata	**76.8**	**70.12**	**83.28**	**77.36**	**89.51**	**74.67**	**97.85**

Tabel A. 25 Rekap performa hasil uji coba skenario 2 SCICA Daubechies 2

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	89.18	94.08	89.22	79.718	90.48	93.24	98.75
2	89.94	94.97	89.22	81.607	91.21	93.96	98.59
3	89.88	94.97	89.36	81.134	90.54	93.84	99.06
4	89.88	94.97	89.36	81.134	90.54	93.84	99.06
5	89.62	94.97	89.85	78.811	90.28	93.76	99.06
Rata-rata	**89.7**	**94.79**	**89.4**	**80.481**	**90.61**	**93.73**	**98.91**

Tabel A. 26 Rekap performa hasil uji coba skenario 2 SCICA Daubechies 4

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	87.25	78.89	96.91	84.67	96.89	83.96	98.38
2	87	78.82	96.39	84.62	96.15	83.94	98.62
3	87.44	79.91	96.65	83.78	95.86	84.46	99.09
4	87.81	80.56	96.73	83.99	96.25	85.15	98.69
5	87	78.93	96.57	83.72	96.36	83.84	98.55
Rata-rata	**87.3**	**79.42**	**96.65**	**84.15**	**96.3**	**84.27**	**98.67**

Tabel A. 27 Rekap performa hasil uji coba skenario 2 SCICA Daubechies 6

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	92.25	91.37	92.81	92.44	95.55	95.22	97.22
2	91.88	90.81	91.9	93.66	95.76	95.43	96.44
3	92.12	91.61	91.9	93.34	95.78	95.33	96.85
4	91.94	90.65	91.92	94.35	95.89	95.53	96.37
5	92.25	91.3	92.36	93.37	95.67	95.32	97.05
Rata-rata	**92.08**	**91.15**	**92.18**	**93.43**	**95.73**	**95.37**	**96.79**

Tabel A. 28 Rekap performa hasil uji coba skenario 2 SCICA Daubechies 8

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	91.18	90.25	94.32	86.88	95.23	92.92	97.9
2	91.44	90.52	94.63	86.75	95.31	92.78	98.22
3	91.12	90.38	94.45	85.97	94.8	92.59	98.38
4	91.12	90.59	94.3	86.02	94.7	92.92	98.21
5	91.38	91.2	94.33	86.09	94.81	92.82	98.53
Rata-rata	**91.25**	**90.59**	**94.41**	**86.34**	**94.97**	**92.8**	**98.25**

Tabel A. 29 Rekap performa hasil uji coba skenario 2 WICA Daubechies 2

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	74.81	94.1	54.64	75.61	69.69	91.45	97.68
2	74.5	93.4	54.82	75.2	69.61	91.56	97.35
3	74.5	93.95	54.9	74.44	69.66	91.36	97.43
4	74.25	93.55	54.43	74.67	69.66	90.98	97.35
5	74.81	94.1	54.64	75.61	69.69	91.45	97.68
Rata-rata	**74.58**	**93.82**	**54.69**	**75.11**	**69.66**	**91.36**	**97.5**

Tabel A. 30 Rekap performa hasil uji coba skenario 2 WICA Daubechies 4

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	80	70.72	89.9	79.16	92.68	75.74	98.65
2	80.38	71.62	90.41	78.44	92.27	76.45	98.9
3	80	70.72	89.9	79.16	92.68	75.74	98.65
4	80.38	71.62	90.41	78.44	92.27	76.45	98.9
5	80.31	71.76	90.28	77.73	92.06	76.38	99.05
Rata-rata	**80.21**	**71.29**	**90.18**	**78.59**	**92.39**	**76.15**	**98.83**

Tabel A. 31 Rekap performa hasil uji coba skenario 2 WICA Daubechies 6

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	82.31	85.22	77.61	86.39	85.35	89.63	96.65
2	82.18	85.32	77.43	85.89	85.15	89.74	96.58
3	82.38	85.01	78.05	86.35	85.37	89.78	96.66
4	82.44	85.43	78.2	85.28	85.56	89.63	96.65
5	82.68	85.28	77.87	87.53	85.48	90.35	96.5
Rata-rata	**82.4**	**85.25**	**77.83**	**86.29**	**85.38**	**89.83**	**96.61**

Tabel A. 32 Rekap performa hasil uji coba skenario 2 WICA Daubechies 8

Percobaan	*Accuracy*	*Sensitivity*			*Specificity*		
		1	**2**	**3**	**1**	**2**	**3**
1	81.31	89.83	70.33	86.42	78.57	92.26	98.52
2	81.75	90.77	70.84	85.17	78.82	92.66	98.58
3	81.18	90.17	69.51	86.44	78.3	92.58	98.35
4	81.5	89.06	71.52	86.12	79.52	91.95	98.28
5	81.38	90.46	69.99	85.85	78.68	92.36	98.44
Rata-rata	**81.425**	**90.06**	**70.44**	**86**	**78.78**	**92.36**	**98.43**

[Halaman ini sengaja dikosongkan]

DAFTAR PUSTAKA

[1] World Health Organization, Oktober 2012. [Online]. Available: http://www.who.int/mediacentre/factsheets/fs999/en/.

[2] "Epilepsy: social consequences and economic aspects," [Online]. Available: http://www.allcountries.org/health/epilepsy_social_consequences_and_economic_aspects.html. [Diakses 3 Juni 2015].

[3] R. Harikumar dan T. Vijayakumar, "Performance Analysis of Patient Specific Elmanchaotic Optimization Model for Fuzzy Based Epilepsy Risk Level Classification from EEG Signals," *International Journal on Smart Sensing and Intelligent Systems, Vol. 2,* p. 612, 2009.

[4] N. Nicolau dan J. Gergiou, "Detection of Epileptic Electroencephalogram Based on Permutation Entropy and Support Vector Machines," 2012.

[5] Riwinoto dan B. Kusumoputro, "Penggunaan Independent Component Analysis (ICA) untuk Pembuangan Noise dan Artefak pada Sinyal Campuran," *National Conference: Design and Application of Technology,* 2010.

[6] W. Zhou dan J. Gotman, "Automatic Removal of Eye Movement Artifacts from the EEG Using ICA and the Dipole Model," *Progress in Natural Science,* vol. 19, p. 1165, 2009.

[7] M. Sheoran, S. Kumar dan A. Kumar, "Wavelet-ICA based Denoising of Electroencephalogram Signal," *International Journal of Information & Computation Technology,* vol. 4, p. 1205, 2014.

[8] B. Mijovic, M. De Vos, I. Gligorijevic, J. Taelman dan S. Van Huffel, "Source Separation From Single-Channel Recordings by Combining Empirical-Mode Decomposition

and Independent Component Analysis," *IEEE Transactions on Biomedical Engineering,* vol. 57, p. 2189, 2010.

[9] P. Jahankhani, V. Kodogiannis dan K. Revett, "EEG Signal Classification Using Wavelet Feature Extraction and Neural Networks," *International Symposium on Modern Computing,* p. 52, 2006.

[10] H. Susanto, "Transformasi Wavelet Haar," 10 Maret 2010. [Online]. Available: http://www.scribd.com/doc/50467423/Transformasi-Wavelet-Haar.

[11] A. Quotb, Y. Bornat dan S. Renaud, "Wavelet transform for real-time detection of action potentials in neural signals," 15 Juli 2011. [Online]. Available: http://journal.frontiersin.org/article/10.3389/fneng.2011.00007/full. [Diakses 3 Juni 2015].

[12] M. Sushama dan G. Tulasi Ram Das, "Detection and Classification of Voltage Sags Using Adaptive Decomposition and Wavelet Transforms," *International Journal of Electrical and Power Engineering,* vol. 3, no. 1, p. 53, 2009.

[13] K. Kawaguchi, "Artificial Neuron with Continuous Characteristics," 17 Juni 2000. [Online]. Available: http://wwwold.ece.utep.edu/research/webfuzzy/docs/kk-thesis/kk-thesis-html/node14.html. [Diakses 3 Juni 2015].

[14] C. W. Dawson, R. L. Wilby, C. Harpham, M. R. Brown, E. Cranston dan E. J. Darby, "Modelling Ranunculus Presence in the Rivers Test and Itchen Using Artificial Neural Networks," [Online]. Available: http://www.geocomputation.org/2000/GC016/Gc016.htm. [Diakses 20 Januari 2015].

[15] T. M. Mitchel, Machine Learning, Singapura: The McGraw-Hill, 1997.

[16] S. Mojarad, S. Dlay, W. Woo dan G. Sherbet, "Breast Cancer Prediction and Cross Validation Using Multilayer Perceptron Neural Networks," *CSNDSP,* p. 760, 2010.

[17] H. J. Hamilton, "Confusion Matrix," 2012 Juni 2012. [Online]. Available: http://www2.cs.uregina.ca/~dbd/cs831/notes/confusion_matrix/confusion_matrix.html.

[18] [Online]. Available: http://www.compumine.com/web/public/newsletter/20071/precision-recall. [Diakses 5 Mei 2015].

[19] "Rencana IT," [Online]. Available: https://rencanait.wordpress.com/2010/03/08/data-mining-data-preprocessing/. [Diakses 25 April 2015].

[20] G. Panchal, A. Ganatra, Y. P Kosta dan D. Panchal, "Behaviour Analysis of Multilayer Perceptron with Multiple Hidden Neurons an Hidden Layer," *International Journal of Computer Theory and Engineering,* Vol. %1 dari %23, No. 2, p. 333, 2011.

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Aditya Bagusmulya, lahir di Meulaboh, pada tanggal 20 Juli 1992. Penulis menempuh pendidikan mulai dari SDS Pawyatan Daha 2 Kediri (1998-2004), SMP Negeri 1 Kediri (2004-2007), SMA Negeri 1 Kediri (2007-2010) dan S1 Teknik Informatika ITS (2011-2015).

Selama masa kuliah, penulis cukup aktif dalam organisasi Himpunan Mahasiswa Teknik Computer-Informatika (HMTC). Diantaranya adalah menjadi staff Departemen Hubungan Luar HMTC ITS 2012-2013 dan Kepala Departemen Hubungan Luar HMTC ITS 2013-2014. Penulis juga aktif dalam berbagai kegiatan kepanitiaan yaitu SCHEMATICS 2012 sebagai staff Keamanan dan Perijinan, SCHEMATICS 2013 sebagai staff Hubungan Masyarakat, ITS EXPO 2012 dan 2013 sebagai staff Workshop Seni. Selama kuliah di Teknik Informatika ITS, penulis mengambil bidang minat Komputasi Cerdas Visi (KCV). Penulis pernah menjadi asisten dosen untuk Program Studi D1 PIKTI ITS mata kuliah Basis Data Terapan dan Excel untuk Bisnis. Komunikasi dengan penulis dapat melalui email: **aditya.bagusmulya@gmail.com**.

[Halaman ini sengaja dikosongkan]